

Water resources systems
Prof. P. P. Mujumdar
Department of Civil Engineering
Indian Institute of science, Bangalore

Model no. # 03

Lecture no. # 14

Introduction to dynamic programming

Good morning and welcome to this lecture number 14 of the course, Water Resources Systems Modeling Techniques and Analysis. Over the last few lectures, we have been discussing about the systems techniques, specifically the optimization techniques. And in the last 3 4 classes, I have been talking about the simplex method for the linear programming problem. So, specifically in the previous lecture, in the last lecture, we discussed how to formulate the dual, given a primal problem and then we also saw that, from the solution of one of the problems, either the primal problem or the dual problem, how to get the solution of the other problem. So, you have the optimal final simplex tableau and then, from the final simplex tableau, you should be able to get the optimal solution of the other problem. If you have solved the primal, you will be able to get the solution for the dual problem.

We also saw the interpretation of the solution of the dual, that is, the dual variables. You can use the dual variables in the sensitivity analysis; specifically we saw the example of the sensitivity analysis, where we are interested in obtaining the change in the objective function for a small change in the right hand side values of the constraints and this is where we use the dual variables, the solution arriving for out of the dual, the problem of the dual f l P and then, we obtain the sensitivity of Z to a change in the right hand side values of the constraints.

So, **that we will** with that, we will formally close the discussion on linear programming, because, we **we** have other techniques to cover, for example, dynamic programming and then, we have the chance constraint linear programming, stochastic dynamic programming, etcetera. Perhaps, what we will do is, we will revisit the linear programming when we were talking about the applications.

So, as of now, I will conclude the discussion on linear programming problem by stating that the other methods of solutions, for example, there are improve methods of solution available for a linear programming, such as for example, the revised simplex method and the carmaker's algorithm **and carmaker's algorithm** and so on, the revised simplex method essentially improve the efficiency of the solutions. In most of the cases, we will have more number of variables then the number of constraints; therefore, the columns in any given simplex will be much larger compared to the number of rows and so on. So, the revised simplex method uses an inversion of the matrix and then increases the efficiency of the solution.

Remember, that most of the problem that we formulated in water resources systems, i has indeed in any engineering problem, engineering field or very large in size and therefore, the efficiency of computational is important. So, the revised simplex method, the dual simplex, the carmakers algorithm etcetera, there are available which will increase the efficiency of a simplex algorithm efficiency of an I P solution, **I am sorry, the simplex algorithm** revised simplex method and dual simplex method. So, deal with, there is also another topic that perhaps will be useful for the water resources systems; students, that has do with a piecewise linear.

(Refer Slide Time: 03: 56)

The slide is titled "Summary of the previous lecture" and contains the following bulleted list:

- Dual problem
 - Formulation of dual problem
 - Dual problem solution from solution of the primal problem
- Sensitivity analysis
 - Change in the RHS of constraints

Below the text, there is a handwritten diagram in red ink showing a piecewise linear function. The function starts at the origin, increases linearly, then curves slightly, and finally levels off. To the right of the diagram, the text "Piecewise Linearization" is written in red. In the bottom left corner of the slide, there is a small circular logo with the text "NPTEL" below it.

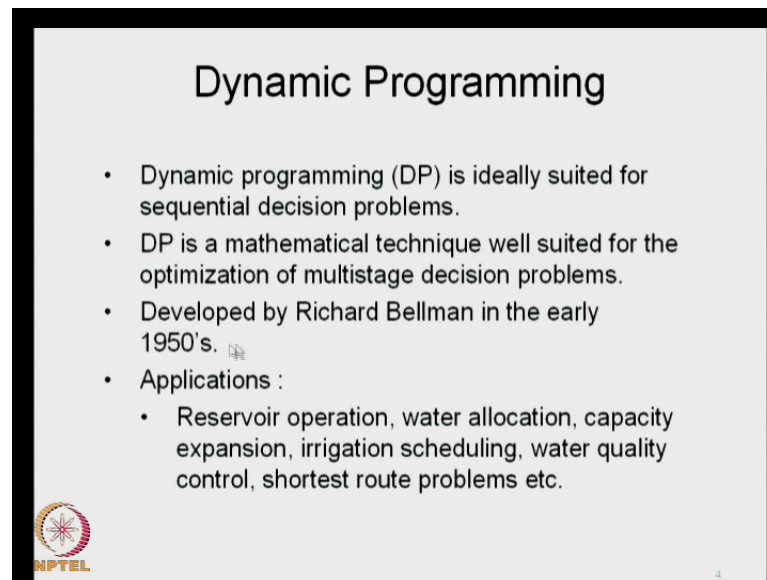
In many situations, we come across problems which have either the objective function or the constraints as non-linear. So, we may not have non-linear objective, **non-linear** we

may not have linear objective function, in which case, it will be difficult to use linear programming problem, in which case, we adopt the piecewise linear problem. Why, we will deal with this when we are talking about actual applications. Suppose, for example, you have a function like this, we can divide this function into several segments and then construct this function as an approximate piecewise linear function. So, this is the general, the actual way of doing it and so on. We will discuss when we are dealing with applications if there is a situation like this, but **to** just to remember at this point, that such techniques are available and also you may have some special cases where you may have only integer numbers, in which case, we convert the linear programming problem into **an** what is called as an integer programming problem and so on. So, there are many variations of the linear programming problem, as I have discussed, are available and we should be able to **use** make use of them. Many of the standard software that are available today for linear programming, we will include all of this, **just keep this remain**, but for the beginning, **students** for the students who are just beginning, the techniques that I have taught here should be adequate to get used to the optimization techniques using linear programming problem.

We will now move on to another important systems technique, another important optimization technique very often used in water resources in analysis **analysis** of the water resources systems. This is the dynamic programming, you know often situation arise especially in water resources where you need to make decisions stage wise. Let us say, you are talking about monthly reservoir operation. So, you release the water for, let us say, hydro power generation, irrigation, municipal and industrial use, etcetera. Month wise **month wise** you are planning, June month you will release so much for irrigation, so much for hydro power and so on. Then, from the storage, you have depleted to another state. So, from one state in beginning of June for example, you went in to another state in the beginning of July next month. Because there was an inflow that was coming in, you made the releases, there are also losses that **I** have taken place and so on. So, the state of the system has changed from one state to another. state in the next month in the example that I have just coated and based on that state of the system, which means, based on the water availability at that particular point in time, you may need to make another decision, how much to be allocated to irrigation, how much to be allocated to water hydro power and so on.


So, depending on the state of the system of which changes from time to time, **to time** you need to take sequential decisions. You make a decision, the state of the system changes based on that state of the system and you have move in time, you make another decision. The state of the system changes again and so on. So, like this, you have to take decisions sequentially one after the other **of after the other** and so on, in an optimal way. Let us, **you have** looking at an yearly reservoir operation and annual **annual** returns in terms of, let us say, hydro power or in terms of the irrigation and so on and within this annual period, you are operating it month wise from June month to July month to August month, etcetera. So, monthly reservoir operation is what you are interested, as an example, this is the typical case of sequential decisions. So, when you are dealing with sequential decisions, all though linear programming problem can be formulated for the sequential decision problems, but a more elegant method of doing it is through dynamic programming problems. So, we will now introduce the dynamic programming problems. The dynamic programming is ideally suited for sequential decisions, when we have situations where the state of the system is changing continuously over time and then, you need to make decisions in an optimal way, in a sequential fashion. The specific type of dynamic programming that we will deal with in this course is the case where we consider only the discrete values for the state as well as for the decisions. I will just indicate that, what we mean by the straight forward, indicate by what we mean by decisions and so on, presently. But, we are dealing with only the case where we will look at only the discrete values of the input state as well as the decisions **alright**. So, we will start introducing dynamic programming. Now, as I said, dynamic programming is ideally suited for sequential decisions and dynamic programming was first formulated by Richard Bellman in the early 50's.

(Refer Slide Time: 09:47)



Dynamic Programming

- Dynamic programming (DP) is ideally suited for sequential decision problems.
- DP is a mathematical technique well suited for the optimization of multistage decision problems.
- Developed by Richard Bellman in the early 1950's.
- Applications :
 - Reservoir operation, water allocation, capacity expansion, irrigation scheduling, water quality control, shortest route problems etc.

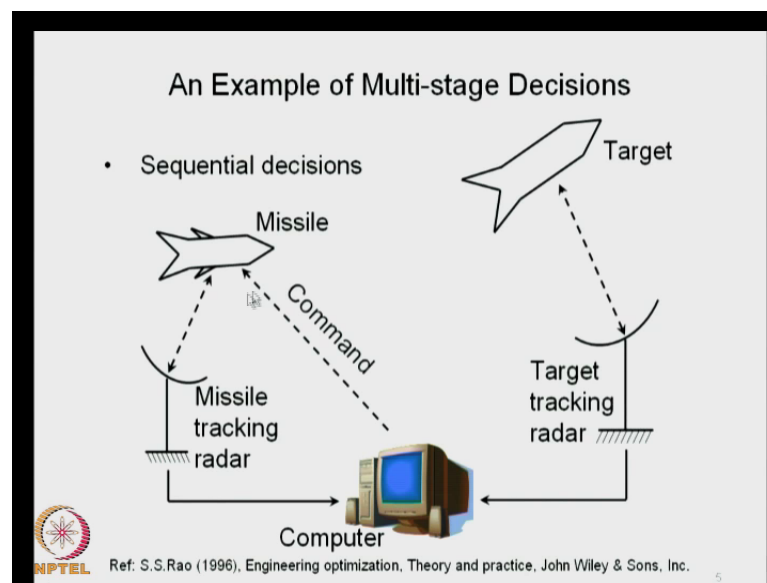
 NPTEL

So, we have associated with dynamic programming, always Bellman comes **comes** out and whether the principle of optimality that I will presently introduce is in fact, called as the Bellman's principle of optimality in water resources.

Dynamic programming is ideally suited for several applications, as I just mentioned, the reservoir operation problem, the classical reservoir problem, which we will deal with in greater details subsequently in the course, **is** can be ideally addressed by dynamic programming. Then, you have water allocation problem, for example, you may want to allocate a given amount of **to** addressing by dynamic programming. Then, you have capacity expansion, where, you have a given infrastructure available, let us say a dam, that has been built, a reservoir irrigation system that has been built and over a period of time next 50 years, also something you would like to plan for expansion of the capacity of the systems, then irrigation scheduling a given amount of water how much to allocate among different crops within the, let us say, 10 day interval intervals and so on. Then, water quality control in the first lecture, I will introduce what is **what is** the water quality control problem. You have several check points and then there is a stream and then you are receiving effluents in the stream. How do we allocate the assimilate capacity of the river in terms of optimal decisions on the treatment and so on. So, this can be converted into a sequential decision problem, when we are talking about decision across different, see **someone see them** one set of treatments non zone, see then another set of treatment and so on and of course, the shortest route problems which are not directly **a directly**

applicable in water **large** water resources problems. But, when we are talking about pipe networks and **and**, so on, the shortest route problems also become relevant. So, these are some of the applications that we often come across in water resources and it is important to understand the dynamic programming and especially at the student level, it is important to understand the applications and small numerical problems which can be solved with.

(Refer Slide Time: 12:29)



So, let us go through the dynamic programming algorithms. This is the classical example that we provide as an illustration of what we mean by a multi stage decision problem. This is from the control theory, **it** typical example from the control theory and this is given in the **s s** book. In fact, for the optimization **technique** techniques that I have covered, I would encourage the students to look at this particular book, extremely well written, well explained algorithms in this book by **s s**. Let us say, you are talking about hitting a target with a missile, now, this is the control theory example.

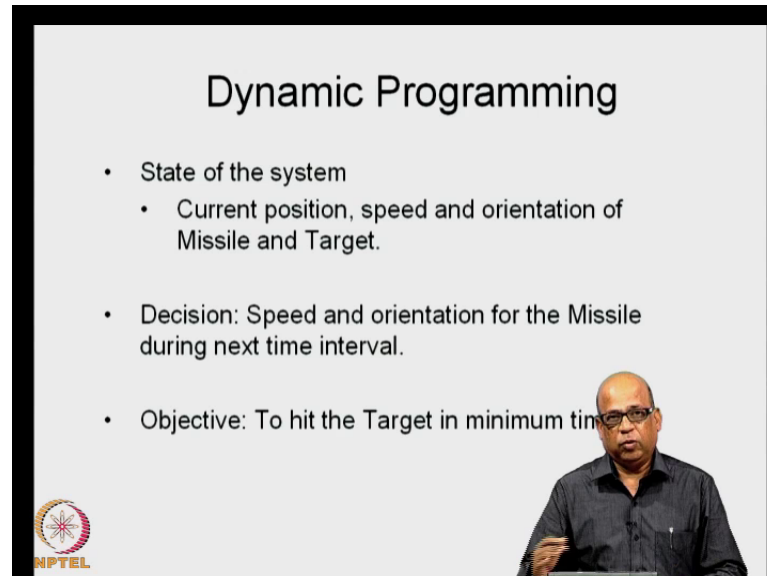
So, I will have to deal with this particular type of example, the missile has to be guided to hit the target in the shortest time. The target has also intelligence sensors, so, it will be sensing at every time interval, time instant, the position and the speed and orientation of the missile and it will take corrective actions to ensure that the missile does not hit the target. There is a target tracking radar which tracks at any instant, the position of the target and the speed and orientation, anything. So, the complete state of the target is

tracked by the radar. So, this is communicated to a computer. Similarly, the missile tracking radar will know at any given point in time, the position, the speed and orientation, etcetera of the missile. So, the complete state of the missile will be tracked by the missile tracking radar. Now, both these information will be coming in real time to a central computer, this will analyze and send a command to the missile saying that you change your orientation, speed, etcetera, in such a manner that you will be able to hit the target in the shortest time possible. So, the command is given to the missile, the missile changes its direction and speed perhaps and several other decisions can be communicated to the command and that will be sensed by the target. Target takes an action, it changes its position, it changes its direction, etcetera, that is tracked by the target **Target** tracking radar. This information comes back to this computer and **the** again, the missile tracking radar, let us say, you have gone 5 seconds, hence, within the 5 seconds, the missile tracking radar again traces the position of the missile and both of these are analyzed together then an optimal decision is communicated to the missile again and then in the next time instant, the missile again changes and so on. So, every time, when we are talking about very small time intervals, here, **in the** in this particular example, every time instant the current position or the current state of the system is analyzed by the computer and then it communicates a decision, optimal decision standing at that particular stage. What do I mean by stage? You are moving from one time step to another time step and each time step here, in this particular example constitutes a stage.

So, from one time step to another time step, you are looking at the current state of the system, what is the state of the system, the position orientation, speed, etcetera of the missile and the position, orientation, speed, etcetera of the target. So, this indicates the current state of the system. Given the current state of the system, you take an optimal decision, for at this stage, such that the chance of this missile hitting the target in a given time, in a given some finite time interval of, let us say 10 minutes is optimized, which means, you would like to hit the target in the shortest time interval. So, this is the classical case where multi stage decisions are taken. You take a decision, the state of the system changes. Based on the state of the system you take another decision such that it is the optimal decision and so on. So, you will revise your optimal decisions as you move from one time step to another time step and so on and finally, when you **when you**

analyze this problem over **given number of states**, given number of stages, **I am sorry**, the decision together will be optimal decision.

(Refer Slide Time: 17:33)



Dynamic Programming

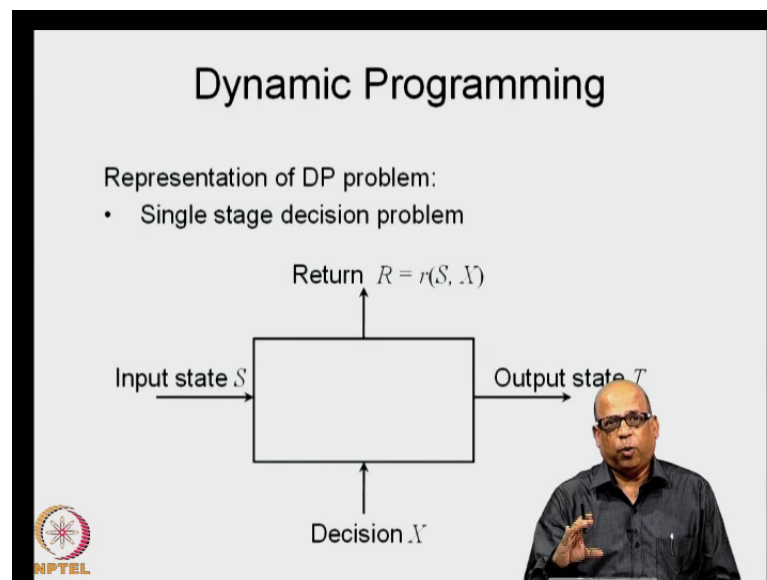
- State of the system
 - Current position, speed and orientation of Missile and Target.
- Decision: Speed and orientation for the Missile during next time interval.
- Objective: To hit the Target in minimum time

NPTEL

So, this is what we mean by a multi stage decision problem. In this particular case, the state of the system as I mentioned, is completely described by the position the speed and the orientation; that means, of the target as well as the missile, these two together and the orientation as well as the speed perhaps and there **there** may be several other things, but I **had** the first graph level. You just understand that these two together, given point in time where the missile is where the target is, what is the orientation of the missile, what is the speed of the missile what is the orientation of the target, what is the speed of the target and so on. So, this becomes **defines** state of the system and from this current state, you move to, **the** you optimize the decisions and you move to the next stage. So, what is the decision that you will take, you will take from the current position? You change your speed and perhaps the orientation, this is the decision that you communicate to the missile and the missile changes it is speed, etcetera, or based on that, what is the objective here? Objective is to hit the target with the missile in the shortest time interval. So, this is the classical multi stage problem. So, the dynamic programming, as we introduced now, will be ideally suited for such multi stage decision problems and multi stage decision problems are very common in water resources systems **al**.

Though the time interval. So, We will be talking about are much larger, typically of order of 10 days, 15 days, one month and so on. So, the control theory time intervals are much much shorter, however, the principles remains the same. So, you understand the principles that we are talking about multi stage decision problems and we are talking about taking optimal decisions, when the state of the system is changing. So, depending on the state of the system at a given time interval or at given stage, stage may need not be always associated with time. Stage may be associated with space, stage may be associated with some abstract quantities and so on. At any given stage, you take an optimal decision such that, something gets optimized over a number of stages; will see what what we mean by that.

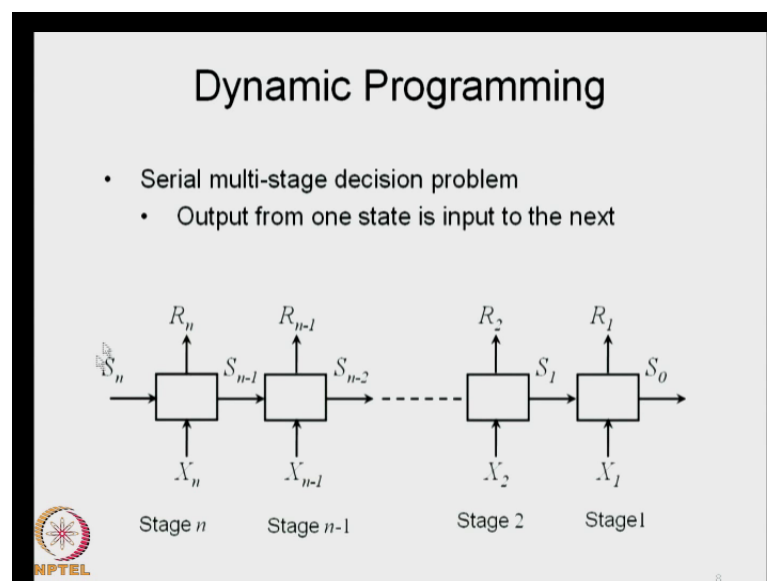
(Refer Slide Time: 20:00)



So, we represent such decision problems with diagrams like this. For example, you may have an Input state s , this is the single stage decision problem in 1 stage. What are all the variables involved or what are all the that we use and so on. So, there is an input state s and you make a decision. I denote it here by capital X to indicate that this can be a vector of decisions $X_1 X_2 X_3$, etcetera. There may be several such small decisions that you may be taking. You make a decision, because of this decision, you get a return. Remember, this return can be either monitor rate and in case of our water resources systems. So, if we are talking about irrigation hydro power and so on or it can be quantitative return in terms of the amount of hydro power generated, amount of flood control that is exercised, amount of irrigation return that you get in terms of the crop

yield and so on. **on** The state you have taken a decision and you get a return because, on this particular state you took a decision, the state of the system changes and it goes to an output state. So, the output state t here, is a function of the Input state s and the decision that you make X , the return here is a function of the Input state s and the return that you get here. For example, you have a reservoir and you **you** are operating the reservoir; what do I mean by operating the reservoir? You are releasing certain amount of water based on the amount of storage that is the available. So, the amount of storage that is available can be a state that becomes an Input state and on this is state you have taken a decision of how much to release from the reservoir during that time period and because of which, you get a return in terms of let to say, the hydro power that you generated during that time period. So, that becomes the return because, from the available storage, you released certain amount of water. The storage changes to a new **new** storage and that becomes the output state. So, the Input state was a storage of the beginning of the time period, the decision is the release that you made during the time period, the output state is the resulting end of the period storage and the return that we mean, we get, is the amount of hydro power as an example. So, this is the single **single** stage decision problem; however, in most engineering problems, the problems are multi stage decision problems.

(Refer Slide Time: 22:56)



What do I mean by that? The multi stage decision problem represented like this, let us say, you are here now and s_n is the input to the state stage n , you make a decision of X


n, you get a return of R_n as the result of which, the state of the system changes and this state, this change in the state becomes the Input state to the next stage n minus one here. You make another decision, you get another return and X_{n-1} and s_{n-1} together will define s_{n-2} and so on until you complete all the stages. So, the output from one state becomes an input to the next and then you would like to optimize the entire system; like this, across all the stages, you would like to optimize. So, this is what is called as a multi stage decision problem and typically in water resources, we get a large number of such problems, reservoir operation is a typical problem, water allocation is another typical problem and so on. So, as we progressed, we will cover several examples dealing with this. So, how do we optimize this? At any given stage, let us say, you are talking, you are studying at any particular stage. Here, you know the Input state, if you know the Input state at that particular point, you will ask what should be my optimal decision such that, something good happens across all the stages put together and this optimization, this multi stage optimization is governed by the Bellman's principle of optimality. We will state it formally and then I will explain the principle of optimality which is commonly called as Bellman's principle of optimality. read size this

(Refer Slide Time: 25:00)

Dynamic Programming

Bellman's principle of optimality:

- "Given the current state of system, the optimal policy (sequence of decisions) for the remaining stages is independent of the policy adopted in the previous stages".
- The principle implies that, given the state S_i of the system at a stage i , one must proceed optimally till the last stage, irrespective of how one arrived at the state S_i .

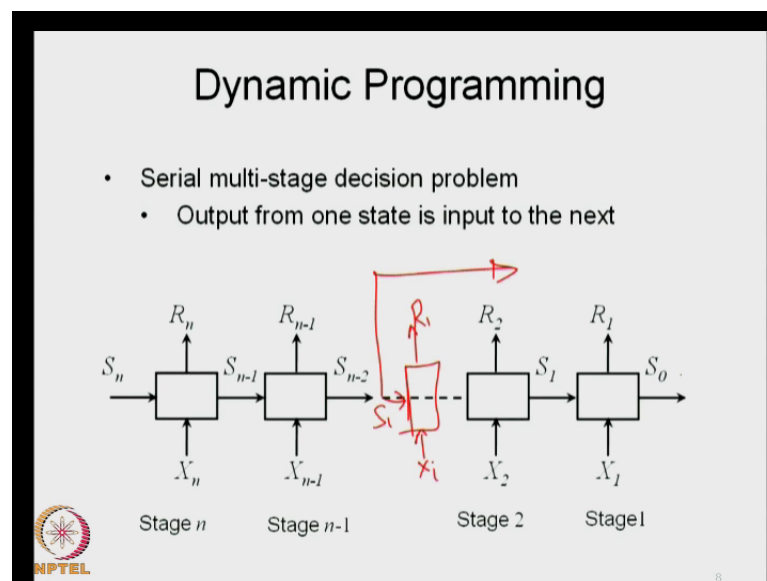


9

Let us read it first and then we will see the explanation. It says, given the current state of the system, the optimal policy or sequence of decisions for the remaining stages is independent of the policy adopted in the previous stages. What does this mean; that

along as long as you know the current state of the system, you do not worry about how we arrived at the particular state. You could have taken several policy decisions and arrived at the particular state, but from this particular point, given the current state, you should proceed optimally until the end of the all the stages. So, this principle implies that given the state s_i of the system at a stage i , one must proceed optimally till the last stage irrespective of how one arrived at the state s_i .

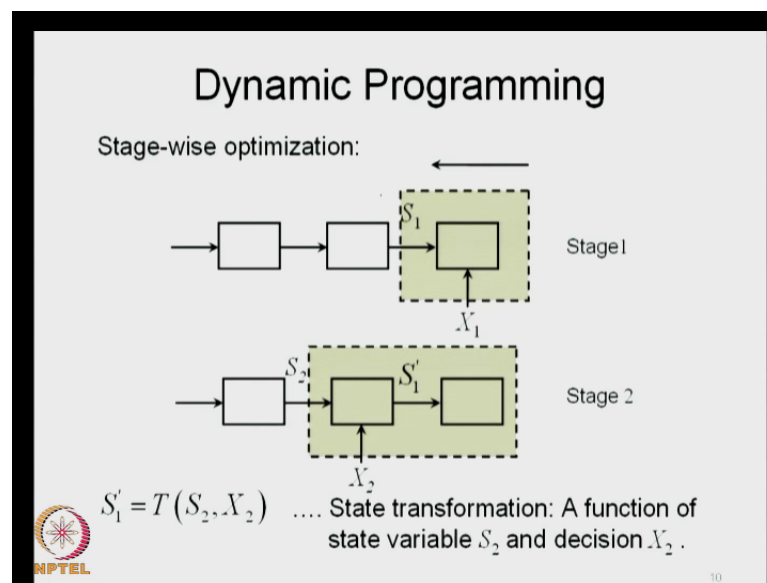
(Refer Slide Time: 26:10)



This is a very simply stated principle, but extremely powerful principle, which makes the computations very easy as I will presently demonstrate. You understand it this way that, at any intermediate stage, let us say, that we are somewhere here a at an intermediate point, you know the stage, will call it as s_i here and you are making a decision X_i and you are getting R_i and so on. Now, if you are moving in this direction, in the forward direction you have arrived at s_i let us say, what the Bellman's principle of optimality states is that, at this is stage, when you are standing in this stage, do not worry about how you arrived at s_i , that is, you would have taken a sequence of decisions here and arrived at s_i , but from this point onwards, you should progress optimally, that is, you are standing at this point and then from this point onwards you proceed optimally is what it states and this is an extremely handy principle which makes the computations very very simple. So, all it says is that, given the state of the system at a particular stage, you have to proceed optimally from that state. do not worry about how you arrived at, you would have taken several decisions earlier to arrive at this point. Given that you are here, you

proceed optimally until the end of **the until the end of** the horizon, whatever horizon is, say, if you are talking about, let us say, annual optimization, etcetera, end of the year. You understand it this way that, at the beginning of, let us say, september month, you know the state of the system and then from **the** september you proceed until May if your year is from June to May. You proceed until May in an optimal manner; do not worry about how we arrived at this september beginning storage and so on. So, as we solve the example, it becomes more and more clear, but, this is the principle that given the state of the system, you proceed in an optimal manner until the end of the horizon.

(Refer Slide Time: 28:37)



Now, how do we apply this principle? let us say you are looking at such a multi stage optimization problem, that is, you have stages like this 1 2 3 etcetera, these can be time stages as I said or space or some abstract users and so on. So, these can be any stage, the definition of stage state variable decisions extra will vary from problem to problem. So, let us say you are in stage 1 here, where you have one block. Given s_1 , you have to make an optimal decision X_1 and you may get a return here you may get **a end of the** end of the stage state and so on and then you make this decision. Given the state s_1 , you make the decision X_1 which transforms the state to s_2 , then you go **go** to transform the state to X_1 in this particular case. Then, you go to the next stage.

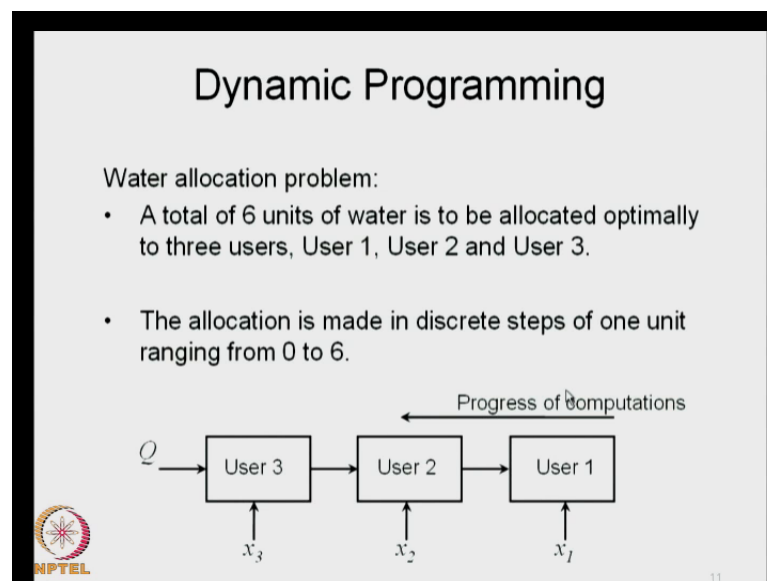
When you go the next stage, you are including two blocks here; that means, what is the decision that you take a on s_2 starting with s_2 . s_2 is the state of the system at the

beginning of the stage 2 at stage 2 and the decision that you take is X_2 . You have taken a decision X_2 , because of which, you get a change with the state of the system s_1 dash, will call it and that becomes an input; but we have already solved for any input that comes into the stage 1. So, s_1 dash defines the input that goes in to stage 1. This we have already optimized and therefore, we know **what is** the optimal return that you get for an optimization associated with state s_1 dash, but the s_1 dash is governed by s_2 and X_2 through a state transformation. So, s_1 dash is given by a state transformation which is the function of s_2 and X_2 . Let us say you are talking about reservoir operation problem, the state of the system here and we are proceeding in the backward direction. Let us see, you standing in June, then, you go to May, then you go to April and so on, so that, time intervals may be defining a stages the state may be storage amount of storage available and so on and the decision may be simply a reservoir released. When I denote it by capital X here, it may mean a vector. Similarly, capital X may mean a vector state variable and so on. So, you first solve for the June month, then you solve for May and June together, then you solve for April, May and June together and so on. So, every time you are every stage, that every stage you are including one time step and then solving it for this 2 together, this 3 together, this 4 together and so on, like this.

So, the Bellman's principle of optimality says, as long as you know this is state, **as long as you know this is state** from this is state, you proceed in an optimal manner. How do I proceed in an optimal manner? In this case, you make a decision X_2 here, thus, a result of **a** which, you will get s_1 dash. But, you know what is the optimal route that you have to take, if you know the state of the system, because, you could have already solved for this. So, like this, you proceed. In **in** this particular case, you know the backward direction. Every time looking until the end of the time on **arisen**, this is the end of the arisen, not necessarily **not necessarily** time, but end of the problem, can imagine, **so**, this is the end of the problem. So, you look always until the end of this arisen and **and** move in an optimal quantum. So, from X_2 you make a decision X_2 , you define the state here and you know what is to be done, because, you already optimized. So, like this, stage wise you move in an optimal manner until you end, until you include all the time steps if **you are stages are in** the stages are defined on the time. Now, we will start with a simple problem to demonstrate what we mean by the principles of optimality, but, before that, I will again repeat **that what we are** the type of problem that we will be dealing with **in the type of problem that will be dealing with** in this course. The state variables are all

discrete; they can take only discrete values. For example, if you are talking about reservoir storage or state variables, it can take let us say, a value of 0, it can take a value of 100, 200, 300 and so on. It cannot take continuous values on a given line. Similarly, the decision variables are all discrete, the decisions that you take are all discrete values. So, this particular problem is called as discrete state dynamic programming problems **alright**.

(Refer Slide Time: 33:59)



Let us look at a simple water allocation problem first. We will solve a simple numerical example, then I will explain the complete principle. **that are a** let us say that we have 6 units of water, may be 6 million cubic meters of water or some something and we want to allocate the 6 units of water in a discrete units among three users. These users can be either a city or different functions. For example, you may have 3 cities and you want to allocate an amount six million cubic meters of water at **in a in a** some time interval. Will not worry in this particular problem, what the time, it is simply, special allocation of water to user 1, user 2 and user 3 or three different users.

Imagine the three users to be, let us say, irrigation hydro power and municipal and industrial water supply. Now, the allocation we will make in discrete units, let us say, 0 1 2 3 4 etcetera, up to 6. So, the allocations are made in discrete units. The problem here is that, how does the water, that is the available 6 units, how much to be given to user 3, how much to be given to user 2, how much to be given to user 1 and we are progressing


our computation in this direction. For this problem to be addressed, we need the returns that we get, for example, if I allocate 0 unit to user number 1, what do I get out of it? What do I get out of that allocation if I allocate two units? What do I get out of similarly, to user 2, similarly **similarly** to user 3. So, we need the return functions associated with a given allocation and perhaps **some** in some problems associated also with the state of the system, we will address more complicated problems later on. But, at this stage, we must understand that if I make a location of a given unit of water to user 1, what do I get back? Similarly, if I allocate a given amount of water to user 2, what do I get back and so on. So, these are called as return function. So, we must know, what is the return that you get out of an allocation that you make to a particular user now.

(Refer Slide Time: 36:42)

Dynamic Programming

- The returns obtained from the users for a given allocation are as follows

Amount of water allocated	Return from		
	User 3 $R_3(x)$	User 2 $R_2(x)$	User 1 $R_1(x)$
0	0	0	0
1	5	5	7
2	8	6	12
3	9	3	15
4	8	-4	16
5	5	-15	15
6	0	-30	12

 NPTEL

These are the return functions for this problems, that we will know the return functions as X follows, let us say, you have an allocation X and then, you have returns from different users. You understand this correctly, what we mean by, that is, because the water is allocated in discrete units like this 0 1 2 3 4 5 6, you can allocate 0 unit to user 3 and you get the return as 0, if you allocate 0 to user 2, you get return of 0, if you allocate as a return to the user, **sorry** if you allocate an amount of 0 to user number 1, you get a return of 0. Similarly, if you allocate 3 to user number 3, you get 9; if you allocate 3 to user number 2, you get 3; if you allocate 3 to user number 1, you get 15 and so on. You have maximum of 6, which means, you can allocate, let us say 2 **two two** 2 plus 2 plus 2 that makes 6. So, a maximum of 6 can be allocated. If you allocate 2, **two** what will you

get? **What will** You get as return 12 plus 6, that is, 18 plus 8, that is 20 6. If you allocate let us say, 3 and 2 and 1, you may get some other return and so on. So, you can, obviously, enumerate exhaustively all possible solutions and then pick up the 1 that leads to a maximum return. But, that is not of **nice way of doing it** because, in practical situations, we will have large number of such discrete allocations possible and therefore, we use the Bellman's principle of optimality and solve it using the dynamic programming. You have three users, you know for a decision that you make, you know what the return that you will get is. So, you start with 1 user, include 2 users, include all the three users and then optimize the whole. So, this is how you proceed in the competition of the dynamic programming.

(Refer Slide Time: 39:04)

Dynamic Programming

Stage 1: $S_1 \rightarrow$ [User 1] $\xrightarrow{R_1}$

$\uparrow x_1$

$$f_1^*(S_1) = \text{Max}[R_1(x_1)]$$

$$0 \leq x_1 \leq S_1$$

$$0 \leq S_1 \leq Q$$

Handwritten notes:
 S_i : Amount of Water available to be allocated to all users included in that stage.
 Total amount available (=6)

S_1 : Amount of water available for allocation to User 1
 x_1 : Amount of water allocated to User 1
 x_1^* : Allocation to User 1, that results in f_1^*
 $f_1^*(S_1)$: Maximum return due to allocation

NPTEL

In the first stage, **we will** we are starting in this backward direction. Here, user 1, user 2, user 3. So, this is the first stage, this will be the second stage and this will be the third stage in which all the users **are in good**. So, typically, you keep this in mind, keep this figure in **this** mind, in the stage number 1, you have one user, in the stage number 2, you have 2 users, in the stage number 3, you have all the three users. So, this will be the way in which you include users at every stage **alright**. So, for the first stage of this problem, you have 1 user, you have s_1 as the state x_1 as the decision R_1 as the return. So, let us understand this state variable and decision variable correctly. The state s_1 is the amount of water available for allocation to user 1. In general, for such a problem, we will define the state s_i for this particular type of problem, allocation to n number of users. We will

define s_i as the amount of water available to be allocated to all users included in that stage. So, this is how we define the state of the system at a stage. **I** now, s_i is called as the state variable, we called it as state variable because, based on the decision that you have taken earlier, etcetera, it gets, it takes on different values and X_1 is called as the decision variable.

We are making a decision based on the state of the system at that particular point and R_1 is called as the return. So, at stage 1, you look at this, you are defining s_1 here. What does this indicate? s_1 is the amount of water available to be allocated to all the users included in that stage. In this stage, how many users are included? There is only one user. Therefore, s_1 is the amount of water available to be allocated to user 1 and X_1 is amount actually allocated to User 1.

So, we have to make a decision on X_1 , such that the return that we get R_1 is maximized. So, we write this as $f_1^* s_1$ is equal to maximize $R_1 s_1$ which means that we are making a decision on X_1 such that, starting with the state s_1 we are getting a maximum value of R_1 of X_1 . In general, R_1 can also be a function of s_1 and X_1 , but, in this particular case, in this particular example that I am dealing with, R_1 is just a function of X_1 . So, this is the relationship that we write $f_1^* s_1$ is the maximum return due to allocation starting at s_1 at the stage s_1 due to allocation, will say, and X_1^* here is the optimal allocation that you can make allocation to user 1, that results in a $f_1^* X_1$; X_1 is the amount of water allocated to User 1. So, just you understand that this particular expression here, what we are saying here is that, we know s_1 , which is the amount of water available to be allocated to the User 1. At this particular stage, we want to make a decision X_1 , which is the amount of water actually allocated to user 1 such that, the return that we get is maximized. Because, we do not know exactly what the value of s_1 is, what we do is, we say s_1 can take on any value between 0 and q where q is the amount of water available, total amount of water available and in this case, it is equal to 6 units. We say that we do not know exactly what s_1 is, but, s_1 , we can examine with all the discrete value between 0 and q , which in this case it is 6, so, s_1 can be 0, for Example, s_1 can be 1, s_1 can be 2 and so on; it can go up to 6. What will be a X_1 ? What will be the range of X_1 ? X_1 can take on values between 0 and s_1 . If, let us say, s_1 is 3, obviously, X_1 cannot be anything more than 3. X_1 can be only between 0 and 3. So, s_1 can take on any value between 0 and 6 X_1 , which is the amount of water

to be **water** allocated at user 1 can go on only up to s_1 . So, this is the relationship that we write for stage number 1.


And this $f_1^*(s_1)$ here, gives the maximum return or the objective function value, optimal objective function value in this case, this is the maximum objective function value at stage 1. Therefore, we will solve for all possible values of s_1 between 0 and q in discrete units and then get associated with any s_1 value, we get the optimal objective function value defined by $f_1^*(s_1)$. So, the objective function value, the optimal objective function value at stage 1 is $f_1^*(s_1)$, it is the function of state variable. The decision that you take is X_1 here. So, you make a decision X_1 on s_1 such that, you get the optimal objective function value $f_1^*(X_1) = f_1^*(s_1)$ and that is the simple maximum R_1 of X_1 in this particular case and this is the stage.

(Refer Slide Time: 46:02)

Dynamic Programming

S_1	x_1	$R_1(x_1)$	$f_1^*(S_1) = \text{Max}[R_1(x_1)]$	x_1^*
0	0	0	0	0
1	0	0	7	1
	1	7		
2	0	0	12	2
	1	7		
	2	12		
3	0	0	15	3
	1	7		
	2	12		
	3	15		

Contd.




So, let us do that. Now, as I said s_1 , which is the amount of water available to be allocated at stage 1 for user number 1, because, there is the only one user can take on values 0 1 2 3 4 5 6 and so on, up to 6. X_1 is the amount of water allocated. So, this is the available amount, this is the allocated amount. So, if s_1 is 0, which is the amount of water available is 0, you cannot give anything other than 0. So, if you give X_1 as 0, the R_1 of s_1 which the return that you get from user number 1 is 0, so, $f_1^*(s_1)$. For every s_1 here, you have to define $f_1^*(s_1)$. So, for $f_1^*(s_1)$, you get a return of 0 because, this maximum among all possible values of $R_1(X_1)$, which is only

1; in this case will be 0 X 1 star is that value of X 1. We should results in f 1 star X 1, that will be 0; you go to s 1 is equal to 1 which means, what available amount of water for allocation at stage 1, is one you can allocate either 0 or 1. If you allocate 0, the return you get is 0. If you allocate 1, the return you get is 7. You are you are looking at this written table and you are here now, are user 1. If you allocate 0, you got a 0 return, if you allocate 1, you got a 7 return, if you allocate 2, you get a return of 12 and so on. So, you are looking at user number 1 and therefore, you are saying, if the water available for allocation is one, I can either allocate 0 or 1. If I allocate, 0 I get a return of 0; if I allocate 7, if I allocate 1, I will get a return of 7 and the maximum among these two allocations is 7. So, that is what you get as f 1 star s 1 and the corresponding allocation is 1. What does it mean? It says that if your amount of water available is 1, you allocate 1, you go to 2 if your amount of water available at stage number 1 is 2, you can either allocate 0 or allocate 1 or allocate 2. If you allocate 0, you get a return of 0, if you allocate 1 you get a return of 7, if you allocate 2 you get a return of 12 and therefore, the amount the maximum.

(Refer Slide Time: 49:07)

Contd.

S_1	x_1	$R_1(x_1)$	$f_1^*(S_1) = \text{Max}[R_1(x_1)]$	x_1^*
4	0	0	16	4
	1	7		
	2	12		
	3	15		
	4	16		
5	0	0	16	4
	1	7		
	2	12		
	3	15		
	4	16		
	5	15		
6	0	0	16	4
	1	7		
	2	12		
	3	15		
	4	16		
	5	15		
	6	12		

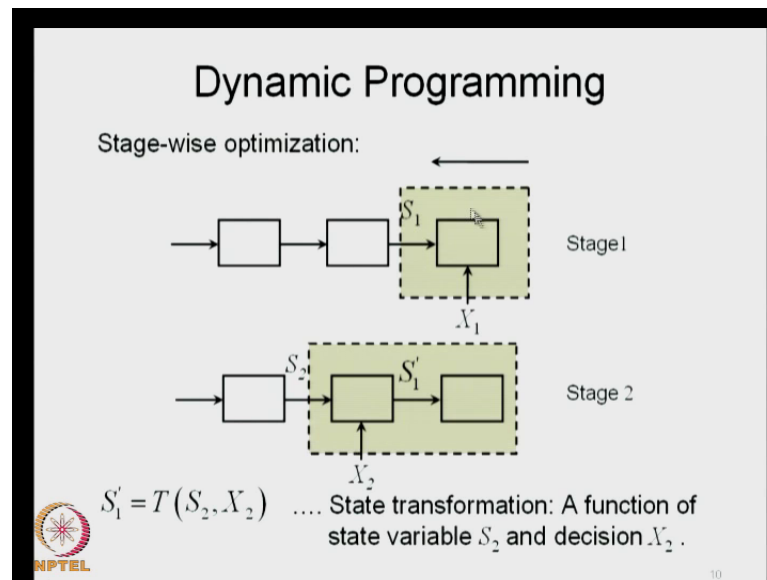


Among these 3, we are looking at the maximum. Among these 3, this is 12, we have look at maximum. Similarly, we go to 3. You can have 0 1 2 3 and the maximum of allocations among all these 3 is 15 and the corresponding allocation is 3. So, in this case, it is saying if 3 is available, you allocate all the 3. Similarly, go to 4. If 4 is available, you will get this kind of returns. You go to 16 and the total amount of water allocated is 4

when you come to 5, that is, its units are 0 1 2 3 4 5, you can allocate up to 5. If you allocate 0, you get a return of 0. Allocate 1 7, allocate 2 12, allocate 3 15, allocate 4 16, allocate 5 15. So, the maximum is again 16, so, you get this as the maximum. So, when 5 is available, it is saying you allocate only 4 to user number 1. Similarly, if 6 is available, it again says you allocate only 4, because, if you allocate more, if there is a penalty here, your returns start **starts** decreasing. So, at stage number 1, what we did is that, we ask the question starting **standing** at the beginning of this is stage. **standing** At this is stage, we know s_1 , what is the amount of water available. We are deciding on X_1 . So, if you know s_1 , what should be my X_1 is the question you are asking and s_1 can be anywhere between 0 to 6 in discrete units. So, we are saying, 0 1 2 3 4 etcetera, up to 6 and then X_1 can take on any value between 0 and X_1 . That is what we did here, X_1 here can take on any value between 0 and 2 0 and 3 here, 0 and 4 here, etcetera. So, X_1 takes on value between 0 and s_1 value here. Associated with each of these allocations, you got the returns and then you are looking at the maximum among all such values of X_1 , that particular value of X_1 which maximize this return $R_1 X_1$. That is what we did here, we are picking up that particular value of X_1 and called it as X_1^* which results in the maximum value of $R_1 X_1$ among all these possible values of $R_1 X_1$. So, at the end of stage number 1 we have answered the question, what should be the allocation to user number 1, if my water available at that stage s_1 is given.

So, if s_1 is equal to 4, what should be my allocation if s_1 is equal to 5, what should be my Allocation? so, this is the question that we have answered; at stage number 1 we do not know exactly what is the available, but we know that X_1 can go on between 0 and 6, it cannot be anything other than 0 and 6. So, we have answered the question if 0 is **the** available, what should be the allocation if 1 is **the** available, what should be the allocation and so on, up to 6 and therefore, when we move to the next stage, we are moving back in the backward direction. In the first stage, we could only 1 user, we move then to the next stage where, we will in put two users. Now, user number 2 and user number 1 together, we understand it this way again, using this diagram in the first stage.

(Refer Slide Time: 52:40)



We solved only for user number 1 for all possible values of s_1 , which was the water available. We now move to stage 2, where we will **in** put two users and we will answer the same question again. These 2 now, is the amount of water available to be allocated at stage 2 for all the users included in that is stage. now there are 2 users, **now** user number 1 and user number 2. The decision that we make at stage 2 is how much to be allocated to user number 2? now, this is the user number 1, user number 2, how much to be allocated to user number 2 such that the total return that you get out of this is the maximum. Because you allocated a certain amount of water to user number 2, you are left with certain other amount of water available s_1 dash and you have already solved for this amount of water available, you have already solved what is the optimal decision that you have to make and therefore, you make a decision on X_2 such that, the total return that you get is maximum. Form here, you **are** proceed in an optimal manner, because, you have already obtained the optimal decisions corresponding to a given state here. So, you know how to move in an optimal manner, here you make a decision X_2 such that, you will end up in its state, so that, together, it is optimized and that is where we use the Bellmen's principle of optimization. If you understand the computation first stage 2, you can include any number of stages later on.

So, we will move to stage 2 in the next lecture. So, the principle that you must have understood is that, standing at any particular state stage, knowing the state at that particular stage, you must move in an optimal manner until the end of **the**. So, we will

continue this discussion in the next lecture. So, in today's lecture, essentially I started introducing the dynamic programming problem. dynamic programming problems are ideally suited for multi stage decision problems and multi stage decision problems are essentially those where given the state of the system, you keep on taking decision in an optimal, you keep moving in the optimal manner; from 1 stage to another stage. Typically, the reservoir operation problems are multi stage problems where starting with the known storage at a particular point in time, you take decisions such that, the decisions across a year, for example, are all optimized. The Bellman's principle of optimality is the guiding principle.

For a dynamic programming problems which essentially states that knowing the state of the systems at a particular stage, you move in an optimal manner until the last stage. do not worry about how we arrived at this particular stage, particular state; as long as you know that you are in a particular state, starting with that is state, you make decisions such that, you move you move in an optimal manner until the end of, until all the states are included, until all the stages are exhausted I am sorry.

So, the stage is a particular time or particular space in this in the water allocation problem that we used, it was the particular user and so on. So, stage is a a point where you make a decision; state defines the state of the system at that particular stage. So, in reservoir operation, for example, the state is the amount of water available or the storage, amount of water available in the storage and the stage is the time interval, let us say, June month, July month and so on. So, in the water allocation problem, stage is a user user number 1, user number 1 and 2 together and so on. So, stage is a stage in the computation and the state is water available at that particular state for allocation and so on. So, definition of a stage, definition of the state variable, definition of the decision variable, etcetera, is problem dependent dependent. So, unlike in linear linear programming where you can write elegant to constraints and so on in algebraic forms, the dynamic programming in the programming you have to understand the problem correctly, define the state variables, define the stage, define the decision variables correctly and discretize them in the in the particular type problems. That why we will be dealing with.

So, dynamic programming is in fact, stated as as much as odd, as it is the signs because, you have to understand the problem, formulate the problem in the structure of the

dynamic programming and then solve them using the **using the** Bellman's principle of optimality. So, it is not algorithmic in the sense that we could solve the linear programming with the simplex algorithm, **that that is** that is the standard algorithm available for all linear programming problems. The dynamic programming problems have to be formulated looking at the physical problem and different **army** programming problems may have to be solved differently. **other** the underlying principle remain as same namely, the principle of optimality, so, **in the** towards the end of the lecture I covered the water allocation problem. I started discussing the water allocation problem; we will go to the next stage which is the second stage in the in the dynamic particular problem and once you understand the **the** second stage correctly, any number of stages can be added, because, every time you are adding, you are relating with what you have solved in the previous stage. So, we will continue this discussion in the next class, where I will complete the water allocation problem and then relocate the Bellman principle of optimality for generalization to any type of problem. So, thanking you for your attention will continued the discussion in next class.