Lecture - 27
Dynamic Programming

Hello students. This is lecture number 27, the title of this lecture is dynamic programming, it comes under special types of LPPs. Now, the dynamic programming as the name indicates, it means that it is dynamic as compared to a static programming problem. It means that it changes with respect to some factor let us say with respect to time or some other factor.

(Refer Slide Time: 01:02)
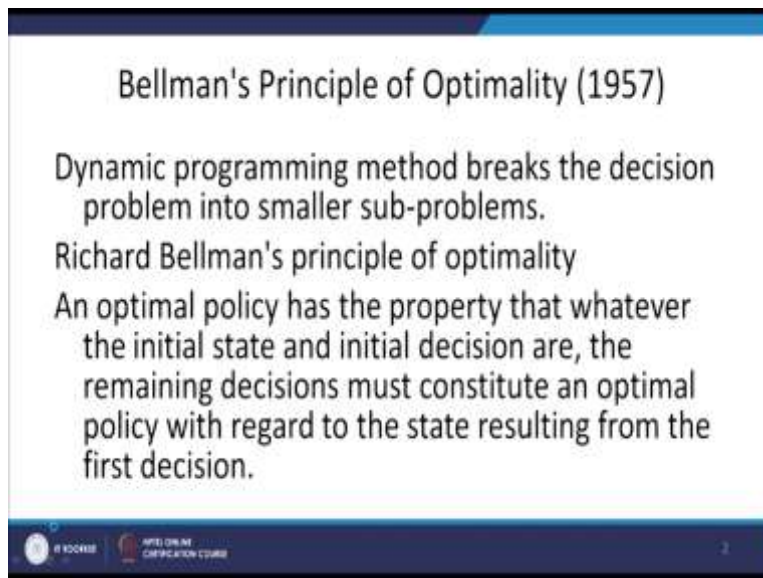
It is also called a **recursive programming problem**. The dynamic programming problem is based on the Bellman's principle of optimality, which was given in the year 1957. Now, this dynamic programming method, it breaks down the decision problem into smaller sub-problems and these sub-problems are solved so as to give the solution of the original problem.

Now, Richard Bellman's principle of optimality is as follows. An optimal policy has the property that whatever the initial state and the initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

(Refer Slide Time: 02:05)

Case 1: Single additive constraint, additively separable return

Find u_j which minimizes

$$z = f_1(u_1) + f_2(u_2) + \dots + f_n(u_n)$$

$$\text{s.t. } a_1 u_1 + a_2 u_2 + \dots + a_n u_n \geq b$$

$$a_j, b \in R; \ a_j \geq 0, \ b > 0.$$

$$u_j \geq 0; \ j = 1, 2, \dots, n$$

Now, let us take some examples to understand what exactly does this mean? So, first of all, let us look at this simple problem, which is the case of a single additive constraint and additive separable returns. Now, this means that suppose u_i's have to be determined in such a way that u_i's are the decision variables which minimizes the function $f_1(u_1) + f_2(u_2) + \dots + f_n(u_n)$. Now, this is a separable function, which means that the function can be written as a sum of different independent variables. For example, $f_1(u_1)$ is a function f_1 and it is only in the variable u_1. Similarly, f_2 is a function which is only in the variable u_2 and like this. So, these kind of functions are called separable functions. Now, this is subject to just one single constraint of the type $a_1 u_1 + a_2 u_2 + \dots + a_n u_n \geq b$. Here, all the a_i's and all the b's, I mean there is only one b, they are all real numbers and they are ≥ 0, a_i's are ≥ 0 and b should be strictly positive. Also the decision parameters u_j's that is $u_j \geq 0$, j goes from 1 to n. So, the decision variables as you know in a linear programming problem, usually the decision parameters are supposed to be ≥ 0.

(Refer Slide Time: 04:01)

The objective or return function *z* is a separable additive function of the *n* variables u_j.

$f_j(u_j)$ is a function of u_j only.

This is an *n*-stage problem, the suffix *j* indicating the stage.

u_j are the decision variables.

With each decision u_j is associated a return function $f_j(u_j)$ which is the return at the *j*th stage.

Now, the objective or the return function z is a separable additive function of n variables u_j's that is $f_j(u_j)$ is a function of u_j only. Just as I have explained and this is an n-stage problem, because why is it n, because the number of decision variables are n and the suffix j indicates the stage. Now, the u_j's are the decision variables with each decision u_j is associated a return function $f_j(u_j)$ which is the return at the jth stage. So, in other words the given problem has been converted into a multi-stage problem like this.

(Refer Slide Time: 04:50)

Introduce **state variables** $x_1, x_2, ..., x_n$ as:
$$x_n = a_1u_1 + a_2u_2 + ... + a_nu_n \geq b$$
$$x_{n-1} = a_1u_1 + a_2u_2 + ... + a_{n-1}u_{n-1} \geq x_n - a_nu_n$$
$$x_{n-2} = a_1u_1 + a_2u_2 + ... + a_{n-2}u_{n-2} \geq x_{n-1} - a_{n-1}u_{n-1}$$
$$...................$$
$$x_1 = a_1u_1 = x_2 - a_2u_2.$$
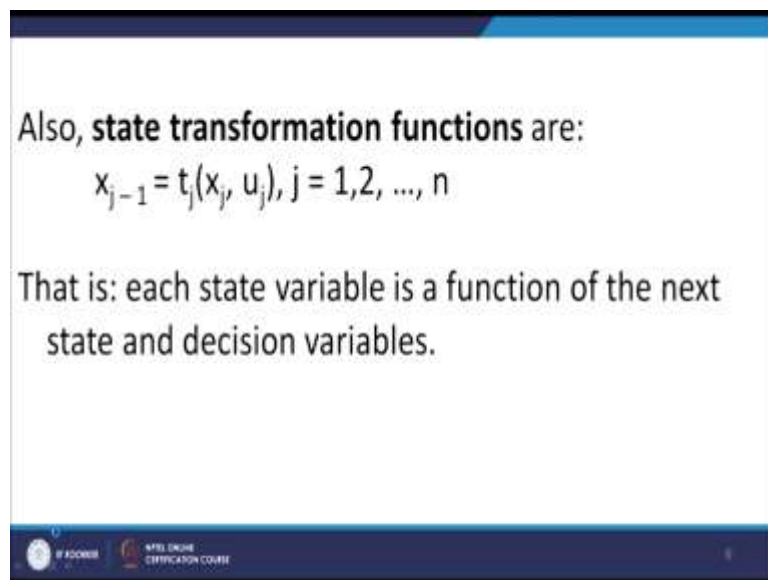
So, let us introduce new variables $x_1, x_2, ..., x_n$ which we will call as the state variables like this. First of all, we will define x_n, which is equal to $a_1u_1 + a_2u_2 + ... + a_nu_n$ and according to the given problem this quantity should be $\geq$ b. Then, we will define $x_{n-1} = a_1u_1 + a_2u_2 + ... + a_{n-1}u_{n-1} \geq x_n - a_nu_n$. This x_n we have taken from the first equation and like this we will go backwards and finally we will define $x_1 = a_1u_1 = x_2 - a_2u_2$.

Now, this is called the backward recursion because we have started by defining from x_n onwards. So, if on the other hand you define starting from x_1 then it is called as a former recursion but since we have just started with x_n, so that is the reason why it is called as a backward recursion.
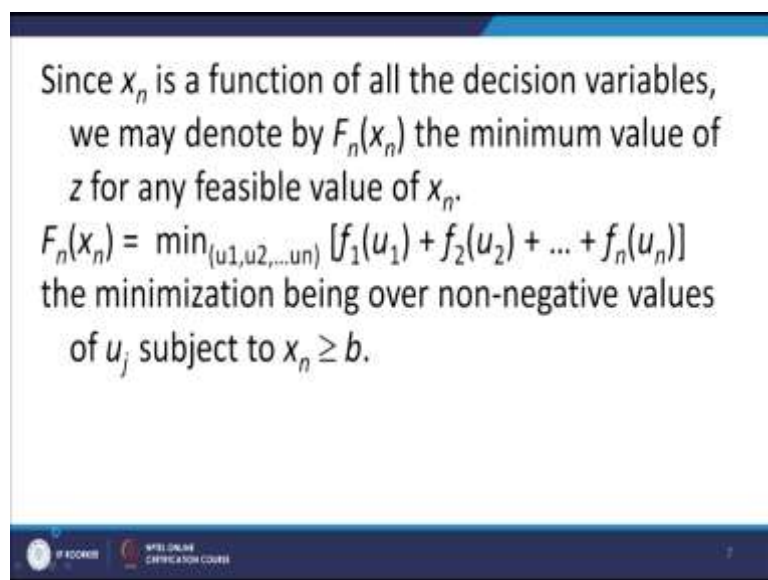
(Refer Slide Time: 06:13)

Also, **state transformation functions** are:
$$x_{j-1} = t_j(x_j, u_j), j = 1, 2, ..., n$$

That is: each state variable is a function of the next state and decision variables.

Now after this, we will also state the transformation functions which are defined like this that is $x_{j-1} = t_j(x_j, u_j)$, $j = 1, 2, ..., n$ and this means that the each state variable is a function of the next state and the decision variables. So, these are called the state transformation functions.

(Refer Slide Time: 06:46)

Since x_n is a function of all the decision variables, we may denote by $F_n(x_n)$ the minimum value of z for any feasible value of x_n.
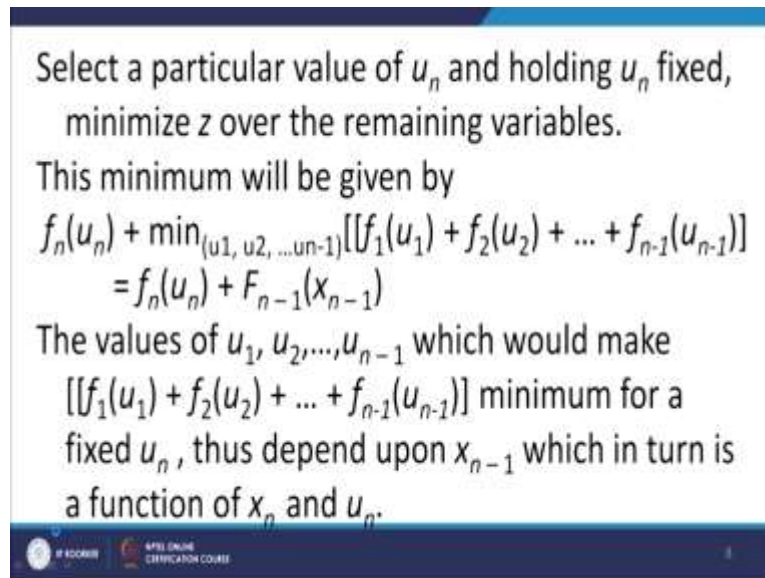$$F_n(x_n) = \min_{(u1, u2, ... un)} [f_1(u_1) + f_2(u_2) + ... + f_n(u_n)]$$
the minimization being over non-negative values of u_j subject to $x_n \geq b$.

Now, since x_n is a function of all the decision variables, we may denote by $F_n(x_n)$ the minimum value of the function z for any feasible value of x_n and we can write $F_n(x_n)$ as $F_n(x_n)$

$= \min_{(u1,u2,...un)} [f_1(u_1) + f_2(u_2) + \ldots + f_n(u_n)]$, and the minimization has to be done over the non-negative values of u_j subject to $x_n \geq b$. So, in other words, we have defined the state transformation variables, we have defined the state variables and the state transformation functions.

(Refer Slide Time: 07:38)

Select a particular value of u_n and holding u_n fixed, minimize z over the remaining variables.
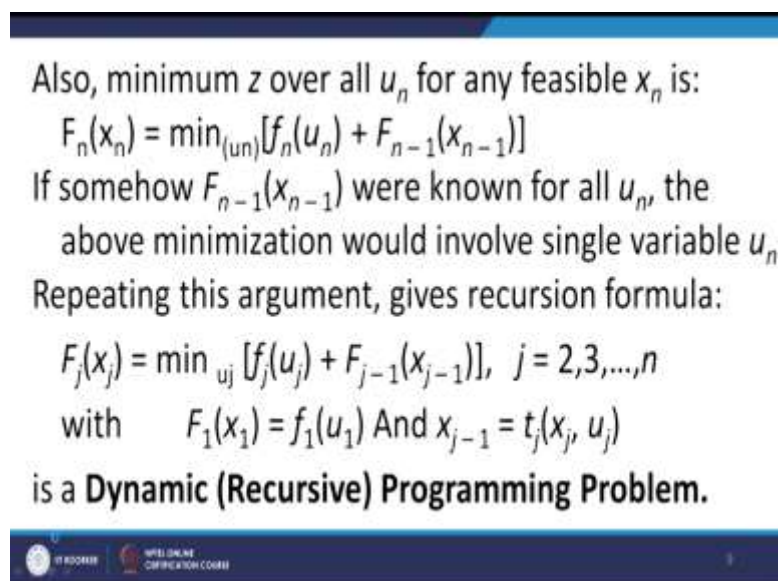
This minimum will be given by

$$f_n(u_n) + \min_{(u1, u2, ...un-1)}[[f_1(u_1) + f_2(u_2) + \ldots + f_{n-1}(u_{n-1})]$$
$$= f_n(u_n) + F_{n-1}(x_{n-1})$$

The values of $u_1, u_2,...,u_{n-1}$ which would make $[[f_1(u_1) + f_2(u_2) + \ldots + f_{n-1}(u_{n-1})]$ minimum for a fixed u_n, thus depend upon x_{n-1} which in turn is a function of x_n and u_n.

Now, we will select a particular value of u_n and holding this u_n fixed, we will minimize the z which is the objective function over the remaining variables and this minimum will be given by $f_n(u_n) + \min_{(u1, u2, ...un-1)}[[f_1(u_1) + f_2(u_2) + \ldots + f_{n-1}(u_{n-1})]$ which is actually equal to $f_n(u_n) + F_{n-1}(x_{n-1})$ and the values of $u_1, u_2,...,u_{n-1}$ which would make this function $[[f_1(u_1) + f_2(u_2) + \ldots + f_{n-1}(u_{n-1})]$ minimum for a fixed u_n. So, this depends upon x_{n-1}, which in fact also in turn depends on the function x_n and u_n. So, that indicates the recursive nature of the problem.

(Refer Slide Time: 08:48)

Also, minimum z over all u_n for any feasible x_n is:

$$F_n(x_n) = \min_{(un)}[f_n(u_n) + F_{n-1}(x_{n-1})]$$

If somehow $F_{n-1}(x_{n-1})$ were known for all u_n, the above minimization would involve single variable u_n

Repeating this argument, gives recursion formula:

$$F_j(x_j) = \min_{uj} [f_j(u_j) + F_{j-1}(x_{j-1})], \quad j = 2,3,...,n$$
$$\text{with} \quad F_1(x_1) = f_1(u_1) \text{ And } x_{j-1} = t_j(x_j, u_j)$$

is a **Dynamic (Recursive) Programming Problem.**

Now also minimum of z over all u_n for any variable for any feasible x_n is $F_n(x_n)$ which is given by $\min_{(u_n)}[f_n(u_n) + F_{n-1}(x_{n-1})]$ and if somehow $F_{n-1}(x_{n-1})$ was known for all u_n, then the above minimization would involve a single variable u_n and like this by repeating this argument, we can get a recursive formula as follows $F_j(x_j) = \min_{u_j} [f_j(u_j) + F_{j-1}(x_{j-1})]$, $j = 2,3,\ldots,n$ with the condition that $F_1(x_1) = f_1(u_1)$ And $x_{j-1} = t_j(x_j, u_j)$. This total concept of defining the state variables and the state transformation functions in a recursive fashion is called as a dynamic programming problem or a recursive programming problem.

(Refer Slide Time: 10:17)

Ex 1: Minimize $u_1^2 + u_2^2 + u_3^2$
Subject to $u_1 + u_2 + u_3 \geq 100$
$u_1, u_2, u_3 \geq 0$
Decision variables are u_1, u_2, u_3
State variables are: $x_3 = u_1 + u_2 + u_3 \geq 100$
$x_2 = u_1 + u_2 = x_3 - u_3$
$x_1 = u_1 = x_2 - u_2$

Now, let us take a simple example to illustrate this procedure. Suppose, we have the optimization problem, $u_1^2 + u_2^2 + u_3^2$ subject to $u_1 + u_2 + u_3 \geq 100$ and the u_1, u_2 and u_3 are $\geq$ 0. So, we will define the decision variables which are the u_1, u_2, u_3. We will define the state variables as $x_3 = u_1 + u_2 + u_3$. This is the first decision state variables x_3 and this condition is already given that it is $\geq$ 100. So, we will write that $x3 \geq 100$. Similarly, going backwards we will define $x_2 = u_1 + u_2$ and this $u_1 + u_2$ is actually equal to $x_3 - u_3$ and finally we will define $x_1 = u_1 = x_2 - u_2$. So, that is the way the state variables are to be defined.

(Refer Slide Time: 11:45)

State transformation functions are:

$$F_3(x_3) = \min_{(u3)}(u_3^2 + F_2(x_2))$$
$$F_2(x_2) = \min_{(u2)}(u_2^2 + F_1(x_1))$$
$$F_1(x_1) = u_1^2$$

Now, $F_1(x_1) = u_1^2 = (x_2 - u_2)^2$

So, $F_2(x_2) = \min_{(u2)}(u_2^2 + (x_2 - u_2)^2)$

To minimize $(u_2^2 + (x_2 - u_2)^2)$ differentiating it w.r.t. u_2

That is: $2u_2 - 2(x_2 - u_2) = 0 \Rightarrow u_2 = x_2/2.$

Hence, $F_2(x_2) = x_2^2/2$

Then, comes the state transformation functions, we are defining them as follows. $F_3(x_3) = \min_{(u3)}(u_3^2 + F_2(x_2))$. This is the definition as I have explained previously and similarly $F_2(x_2) = \min_{(u2)}(u_2^2 + F_1(x_1))$ and finally $F_1(x_1) = u_1^2$. This is possible remember because in the problem that is given to us; note that each of these three terms are in one variable only. That is this first term is in u_1, similarly the second term is in u_2, the third term is in u_3. So, each of the three terms are in different variables and as per the definition, this is called as a separable function. So, now, once we have defined the state variables and the state transformation function, now let us proceed to find out what this $F_1(x_1)$ is and as you know that from this condition $F_1(x_1) = u_1^2$.

And now we will substitute the value of u_1, which is equal to $x_2 - u_2$) and that is the square has also to be taken. So, this means that $F_2(x_2) = \min_{(u2)}(u_2^2 + (x_2 - u_2)^2)$ and now we need to minimize this quantity which is given inside the bracket that is $\min_{(u2)}(u_2^2 + (x_2 - u_2)^2)$ and in order to minimize it, we will differentiate it with respect to u_2 which will give us this condition, $2u_2 - 2(x_2 - u_2) = 0 \Rightarrow u_2 = x_2/2$. So here we have this condition, $u_2 = x_2/2$. Hence, $F_2(x_2) = x_2^2/2$.

(Refer Slide Time: 14:11)

Now, $F_3(x_3) = \min_{(u_3)}(u_3^2 + F_2(x_2))$
$= \min_{(u_3)}(u_3^2 + x_2^2/2)$ $x_2 = x_3 - u_3$
$= \min_{(u_3)}(u_3^2 + (x_3 - u_3)^2/2)$

For minimum $2u_3 - (x_3 - u_3) = 0 \Rightarrow u_3 = x_3/3$

Hence $F_3(x_3) = x_3^2/3$, $x_3 \geq 100$.

Obviously $F_3(x_3)$ is the least for $x_3 = 100$.

So minimum value of $u_1^2 + u_2^2 + u_3^2$ is 1000 100

And $u_1 = u_2 = u_3 = 100/3$

Similarly, $F_3(x_3) = \min_{(u_3)}(u_3^2 + F_2(x_2))$ and we will now substitute this, in place of this $F_2(x_2)$, we will substitute $x_2^2/2$ which we have just obtained and we will get x_2 in terms of (x_3-u_3) because here we will substitute $x_2=x_3-u_3$ and again for minimization, we will have to take its derivative with respect to u_3 and put it equal to 0. So that gives us $u_3 = x_3/3$. Hence, $F_3(x_3) = x_3^2/3$. Now we know that $x_3 \geq 100$, this is way that has been given in the problem. So $F_3(x_3)$ is the least for $x_3 = 100$. So the minimum value of $u_1^2 + u_2^2 + u_3^2$ is 1000 and automatically by going backward, you can see that $u_1 = u_2 = u_3 = 100/3$. So that is the solution of this problem.

(Refer Slide Time: 15:47)

Ex 2: A student has to take examination in three courses, A, B, C. He has three days available for study. He feels it would be best to devote a whole day to the study of the same course, so that he may study a course for one day, two days, or three days or not at all. His estimates of the grades he may get by the study are as follows:

Next, let us take another example which is slightly different in structure. This problem says that a student has to take an examination in three courses A, B and C and he has three days available for study. He feels it would be best to devote a whole day to the study of the same

course, so that he may study a course for one day, two days or three days or not at all. His estimates of the grades he may get by the study are as follows.

(Refer Slide Time: 16:31)

Course / Study days	A	B	C	
0	0	1	0	
1	1	1	1	estimated grades
2	1	3	3	
3	3	4	3	

How should he study so that he maximizes the sum of his grades?

In this table, we have the courses A, B, C and the study days that is 0 1 2 3. For example, if he studies study days are 2 then A 1 hour, B 3 hours and C 3 hours. So, how should he study so that he maximizes the sum of his grades? So, in this table the estimation of his grades is given that is in the courses A, B and C he has if he puts study days 0 1 2 3 then his estimated grades are given by if he studies 0, I mean if he does not study for any day, then in A subject he will get 0 grades and in B he will get 1 grade and in C he will get 0 grade. So, these are the grades, his estimated grades. So, the problem says that how should he study so that he maximizes the sum of his grades that is the problem. So, how should he study so that he maximizes the sum of his grades? That is how many hours should be put in for each of the three subjects or how many days he should put in for each of the three subjects so that his overall grades are maximized.

(Refer Slide Time: 18:07)

Let u_1, u_2, u_3 be the number of days he decides to study the courses A, B, C respectively.

Let $f_1(u_1)$, $f_2(u_2)$, $f_3(u_3)$ be the grades earned by such a study.

The problem is to Maximize $f_1(u_1) + f_2(u_2) + f_3(u_3)$

subject to $u_1 + u_2 + u_3 \leq 3$

   $u_1, u_2, u_3 \geq 0$ and integers

So, let us formulate this problem using the dynamic programming problem that we have learnt. So, let u_1, u_2, u_3 be the number of days he decides to study the courses A, B and C respectively. So let $f_1(u_1)$, $f_2(u_2)$, $f_3(u_3)$ be the grades earned by such a study. So, with this definition, the problem can be written as maximize $f_1(u_1) + f_2(u_2) + f_3(u_3)$ subject to the condition $u_1 + u_2 + u_3 \leq 3$ and all the u_1, u_2, $u_3 \geq 0$ and integers.

Now, the reason why I have chosen this problem is that number one this is a discrete problem because it is talking about integer variables as you can see u_1, u_2, u_3 should be integers and secondly the constraint that has been given is of the less than or equal to type. So, with these two different characteristics, let us look at the solution of the problem.

(Refer Slide Time: 19:24)

Decision variables are u_j

Return functions are $f_j(u_j)$, $j = 1, 2, 3$.

State variables x_j defined as follows.

$$x_3 = u_1 + u_2 + u_3 \leq 3$$
$$x_2 = u_1 + u_2 \qquad = x_3 - u_3$$
$$x_1 = u_1 \qquad = x_2 - u_2$$

State transformation functions are:

$$x_{j-1} = t_j(x_j, u_j), \quad j = 2,3$$

Now, the decision variables are u_j's and the return functions are $f_j(u_j)$, $j = 1, 2, 3$. So, the state variables can be defined as follows, $x_3 = u_1 + u_2 + u_3 \leq 3$. Similarly, $x_2 = u_1 + u_2$ which can be written in terms of x_3 as $x_3 - u_3$. Similarly, $x_1 = u_1$ which can be written in terms of $x2$ and u_2 as $x_2 - u_2$ and also we can define the state transformation functions $x_{j-1} = t_j(x_j, u_j)$, $j = 2,3$.

(Refer Slide Time: 20:21)

Recursive formulae are:

$F_j(x_j) = \max_{u_j} [f_j(u_j) + F_{j-1}(x_{j-1})], \quad j = 2,3$

$F_1(x_1) = f_1(u_1),$

Where $F_3(x_3) = \max_{u_1\ u_2\ u_3} [f_1(u_1) + f_2(u_2) + f_3(u_3)]$

for any feasible x_3.

The required solution will be $\max_{u_j} F_3(x_3)$

Now, this recursive formulas are $F_j(x_j) = \max_{u_j} [f_j(u_j) + F_{j-1}(x_{j-1})], \quad j = 2,3$ and for F_1 it is $F_1(x_1) = f_1(u_1)$, where $F_3(x_3) = \max_{u_1\ u_2\ u_3} [f_1(u_1) + f_2(u_2) + f_3(u_3)]$ for any feasible x_3. The required solution will be $\max_{u_j} F_3(x_3)$. This is using the way in which we have defined the state variables and the state transformation functions.

(Refer Slide Time: 21:11)

Stage returns $f_j(u_j)$

u_j j	0	1	2	3
1	0	1	1	3
2	1	1	3	4
3	0	1	3	3

Now, the stage returns can be written like this. On the top, we have u_j that is 0 1 and 2 3 and on the left hand column we have j which goes from 1, 2 up to 3 and the state returns can be written as follows 0 1 1 3 1 1 3 4 0 1 3 3. This is based upon the stage returns as given in the problem.

(Refer Slide Time: 21:47)

State transformations x_{j-1}, $j = 2,3$

u_j / x_j	0	1	2	3
0	0	(–)	(–)	(–)
1	1	0	(–)	(–)
2	2	1	0	(–)
3	3	2	1	0

Also, we can define the state transformations as x_{j-1} where j goes from 2 to 3 like this. In the top, we have u_j that is 0 1 2 3 and on the left-hand side in the first column we have the values of x_j as 0 1 2 3 and corresponding entries are shown in the table. Please note that some entries are not defined. So, that is the reason why they have been shown with the dashed sign.

(Refer Slide Time: 22:26)

Recursive operations

u_2 / x_2	$f_2(u_2)$				$F_1(x_1)$				$f_2(u_2)+F_1(x_1)$				$F_2(x_2)$
	0	1	2	3	0	1	2	3	0	1	2	3	
0	1	–	–	–	0	–	–	–	1	–	–	–	1
1	(1)	1	–	–	(1)	0	–	–	(2)	1	–	–	(2)
2	1	1	3	–	1	1	0	–	2	2	3	–	3
3	1	1	3	4	3	1	1	0	4	2	4	4	4

Now, coming to the recursive operations, let us try to understand this table how it is written. In the first column, we have the x_2 values that is 0 1 2 3 and in the top rows we have $f_2(u_2)$.

So, $f_2(u_2)$ will depend upon the value of u_2 that is u_2 is 0 1 2 3 and at the second part of the table we have $F_1(x_1)$ and finally in the third part we have $f_2(u_2)+F_1(x_1)$ and the last column is the $F_2(x_2)$.

So from this table, we have to obtain the values of $F_2(x_2)$. Now, this is depending upon the way in which $f_2(u_2)$ and $F_1(x_1)$ is defined and this third is the sum, so this is the sum of the two right and whatever you get that is the $F_{2(x_2)}$ is written. So please note, for example, this 1 I have marked in red and this 1, So 1+1 is 2 so therefore this 2 comes over here and like this all the entries can be verified.

(Refer Slide Time: 23:50)

	$f_3(u_3)$				$F_2(x_2)$				$f_3(u_3)+F_2(x_2)$				$F_3(x_3)$
u_3	0	1	2	3	0	1	2	3	0	1	2	3	
x_3													
0	0	–	–	–	1	–	–	–	1	–	–	–	1
1	0	1	–	–	2	1	–	–	2	2	–	–	2
2	0	1	3	–	3	2	1	–	3	3	4	–	4
3	0	1	3	3	4	3	2	1	4	4	5	4	5

Then coming to the next stage that is $f_3(u_3)$ exactly in the same manner $f_3(u_3)$ is written in the first part of the table. In the second part of the table, we have written $F_2(x_2)$, then in the third part is their sum and finally $F_3(x_3)$ is written in the last. So, here also you can just check these values, for example, this 3 and 2, so 3+2 is 5 and this 5 comes here. So this 3+2 is 5 which comes here and that is the value of our $F_3(x_3)$.

(Refer Slide Time: 24:34)

The maximum value is 5, and tracing the path backwards through bold type numbers we get the optimal policy as $u_3 = 2$, $u_2 = 0$, $u_1 = 1$. In other words for the best cumulative grade point he should study subject A for one day only, subject C for two days and should not devote any time to the study of subject B.

So, the maximum value is 5. How is this maximum value? Look at these entries, so the maximum value is 5 and if you trace the path backwards through the bold type numbers that are marked in red, we get the optimum policy as $u_3=2$, $u_2=0$, $u_1=1$. Just let us go back here and see, look at the tables over here. So, this tells us that this 5 from where did it come? It came from these red entries.

And that is the reason why we get the value of our u_3 as 2, u_2 as 0, u_1 as 1. So, this means that in other words for the best cumulative grade point, the student should study subject A for 1 day only, subject C for 2 days and should not devote any time to this subject B.

(Refer Slide Time: 25:45)

Single multiplicative constraint, additively separable return

Minimize $z = f_1(u_1) + f_2(u_2) + \dots + f_n(u_n)$

s.t. $u_1 u_2 \dots u_n \geq k > 0$, $u_j \geq 0$.

State variables x_j are: $x_n = u_n u_{n-1} \dots u_2 u_1 \geq k$

$x_{n-1} = x_n/u_n = u_{n-1} \dots u_2 u_1$

....

$x_2 = x_3/u_3 = u_2 u_1$

$x_1 = x_2/u_2 = u_1$

So, that is the way this discrete problem has been dealt with. Now, let us look at another situation where we have a single multiplicative constraint and additive separable written. The problem is to minimize z given by $f_1(u_1) + f_2(u_2) + \dots + f_n(u_n)$ subject to $u_1 u_2 \dots u_n \geq k > 0$,

and $u_j \geq 0$. Now, please note that this constraint is a multiplicative constraint that is it is a product of $u_1\ u_2\ldots u_n$ and for this reason we have to define the state variables in a slightly different fashion.

So, the state variables x_j can be defined like this, $x_n = u_n u_{n-1} \ldots u_2 u_1 \geq k$. Similarly, $x_{n-1} = x_n/u_n = u_{n-1} \ldots u_2 u_1$ and finally x_2 and x_1 just as the way we have defined previously.

(Refer Slide Time: 27:03)

State transformations of the type $x_{j-1} = t_j(x_j, u_j)$.
Denoting by $F_n(x_n)$ the minimum value of the objective function for any feasible x_n, we get, the recursion formula
$$F_j(x_j) = \min_{(u_j)} [f_j(u_j) + F_{j-1}(x_{j-1})],\ j = 2,3,\ldots,n$$
With $F_1(x_1) = f_1(u_1)$

Then, we need to define the state transformation functions of the type $x_{j-1} = t_j(x_j,\ u_j)$ denoting by $F_n(x_n)$ the minimum value of the objective function for any feasible x_n, we get the recursive formula $F_j(x_j) = \min_{(u_j)} [f_j(u_j) + F_{j-1}(x_{j-1})]$, $j = 2,3,\ldots,n$ and $F_1(x_1) = f_1(u_1)$. So, exactly just as in the first example, the only difference here is that where we have the constraint is in terms of product instead of the sum.

(Refer Slide Time: 27:52)

Ex: Maximize $u_1^2 + u_2^2 + u_3^2$
subject to $u_1 u_2 u_3 \le 6$, where u_1, u_2, u_3 are positive integers.

Sol: State variables are:
$$x_3 = u_1 u_2 u_3 \le 6$$
$$x_2 = x_3/u_3 = u_1 u_2$$
$$x_1 = x_2/u_2 = u_1$$

So, let us take an example to understand this. Let us take maximization of $u_1^2 + u_2^2 + u_3^2$ subject to the product of $u_1 u_2 u_3 \le 6$ where each of the variables u_1, u_2, u_3 are positive integers. So as before, the state variables are defined as $x_3 = u_1 u_2 u_3 \le 6$, $x_2 = x_3/u_3 = u_1 u_2$ and $x_1 = x_2/u_2 = u_1$.

(Refer Slide Time: 28:34)

Stage returns $f_j(u_j) = u_j^2$, $j = 1,2,3$

u_j	1	2	3	4	5	6
$f_j(u_j)$	1	4	9	16	25	36

Again, the state returns that is $f_j(u_j)$ can be written as u_j^2 where j goes from 1, 2 up to 3 and this is the data that is given 1 2 3 4 5 6. So, it is their squares. Remember $f_j(u_j)$ is u_j^2 so the square of 1 is 1, square of 2 is 4, square of 3 is 9 and like this.

(Refer Slide Time: 28:59)

State transformations $x_{j-1}, j = 2,3$

x_j \ u_j	1	2	3	4	5	6
1	1	–	–	–	–	–
2	2	1	–	–	–	–
3	3	–	1	–	–	–
4	4	2	–	1	–	–
5	5	–	–	–	1	–
6	6	3	2	–	–	1

Then, comes the state transformation function that is the x_{j-1} where j goes from 2 to 3. Here again in the table, u_j is represented at the top row that is 1 2 3 4 5 6 and x_j are shown in the column in the first column and the corresponding values that are shown here in the table and those which are not defined, they are shown with the help of a dashed line.

(Refer Slide Time: 29:31)

Recursive operations

x_1	1	2	3	4	5	6
$F_1(x_1)$	1	4	9	16	25	36

So, the recursive operations are shown here. The x_1 and $F_1(x_1)$ for 1 2 3 4 5 6, this is given in the problem.

(Refer Slide Time: 29:43)

	$f_2(u_2)$						$F_1(x_1)$						$F_2(x_2)$
u_2 x_2	1	2	3	4	5	6	1	2	3	4	5	6	
1	1*	–	–	–	–	–	1*	–	–	–	–	–	2*
2	1	4	–	–	–	–	4	1	–	–	–	–	5
3	1	–	9	–	–	–	9	–	1	–	–	–	10
4	1	4	–	16	–	–	16	4	–	1	–	–	17
5	1	–	–	–	25	–	25	–	–	–	1	–	26
6	1	4	9	–		36	36	9	4	–	–	1	37

And the table this table illustrates the value of $f_2(u_2)$ and the value of $F_1(x_1)$ and $F_2(x_2)$. So, here we find, you can just verify these values that are shown in the table entries and those which are not defined are shown with the help of the dashed line.

(Refer Slide Time: 30:10)

	$f_3(u_3)$						$F_2(x_2)$						$F_3(x_3)$
u_3 x_3	1	2	3	4	5	6	1	2	3	4	5	6	
1	1	–	–	–	–	–	2	–	–	–	–	–	3
2	1	4	–	–	–	–	5	2	–	–	–	–	6
3	1	–	9	–	–	–	10	–	2	–	–	–	11
4	1	4	–	16	–	–	17	5	–	2	–	–	18
5	1	–	–	–	25	–	26	–	–	–	2	–	27
6	1	4	9	–	–	36*	37	10	5	–	–	2*	38*

In the second table, we have the $f_3(u_3)$ and the corresponding $f_2(u_2)$ and the $F_2(x_2)$.

(Refer Slide Time: 30:18)

Maximum value of $u_1^2 + u_2^2 + u_3^2$ is therefore 38
with $u_3 = 1$, $u_2 = 1$, $u_1 = 6$.
(Traced back with a *entry in red in previous tables)
Alternate solution $u_3 = 6$, $u_1 = u_2 = 1$ and
$u_3 = 1$, $u_2 = 6$, $u_1 = 1$ are also possible.
(First of these is traced back with a * entry in red.)

So, the maximum value of $u_1^2 + u_2^2 + u_3^2$ is 38, which is given by the values $u_3 = 1$, $u_2 = 1$, $u_1 = 6$ and from where does this 38 come from? So, this is the 38. So, let us look at by tracing back. So, this is the place from where they have come, so they are shown in the red entries over here in moving backwards and the corresponding to these red entries and therefore we can say that $u_3 = 1$, $u_2 = 1$, $u_1 = 6$. So, these are traced back from, that is why it is called as a backward recursion and therefore we can say that the solution is obtained as $u_3 = 1$, $u_2 = 1$, $u_1 = 6$. The alternate solution that is $u_3=6$ and u_1 and $u_2 =1$ and $u_3=1$, $u_2=6$, $u_1=1$ are also possible. First of these is shown in the tables that I have marked with the red entry. So, in this way, we can solve this problem using dynamic programming.

So, with this we come to an end of this lecture on dynamic programming where we have studied the principle of optimality and we have seen some examples how we can use them to solve the dynamic programming problem. Thank you.