[Academic Script]

Subject	**:**	**Business Economics**
Course	**:**	**B.A., 4th Semester, Undergraduate**
Paper No.	**:**	**403**
Paper Title	**:**	**Quantitative Techniques for Management**
Unit No.	**:**	**3**
& Title	**:**	**Dynamic Programming**
Lecture No.	**:**	**1 (One)**
& Title	**:**	**Dynamic Programming**

Credits

Subject Expert

Dr. Rachna Gandhi

Asst. Prof.

K. S. School of Business Management,

Gujarat University,

Ahmedabad.

Subject Co-Ordinator

Dr. V. Chari

Professor, S. D. School of Commerce,

Gujarat University,

Ahmedabad.

Editing

Jaydip Gadhvi

General Asst.

Jagdish Jadeja

Multimedia

Gaurang Sondarva

Camera

Mukesh Soni

Technician

Mukesh Soni

Technical Assistants

Smita Bhatt

Archana Patel

Helper

Ambalal Thakore

Graphic Artist / Animator

Dilip Dave

Jaydip Gadhvi

Production Assistant & Editing Concept

Mukesh Soni

Producer

Dinesh Goswami

1. Introduction.

Many decision making problems involve a process that takes place in several stages. Here at each stage, the process is dependent on the strategy chosen. Thus dynamic programming is a method of solving problem based on the theory of multistage decision process, in which a sequence of interrelated decisions has to be made.
Mathematically, a DPP is a decision making problem in n-variables, the problem being sub-divided into n sub-problems each sub-problem being a decision – making problem in one variable only. Then the solution of the original problem is achieved by integrating the solutions of sub-problems. Thus it is a systematic, complete enumeration technique.

2. Characteristics of Dynamic programming.

The basic features which characterize the dynamic programming problem are as follows:

Stage i:

Each sub problem of the original problem is known as stage i.

Alternative m_i:

In a given stage i, there may be more than one choice of carrying out a test. Each choice is known as alternative m_i.

Recursive function $f_i(x_i)$:

A function which links the measure of performance of interest of the current stage with the cumulative measure of performance of the previous stages/succeeding stages as a function of the state variable of current stage is known as the recurrence function of the current stage.

Let

$f_1(x_1) = \max[R(m_1)]$

$f_i(x_i) = \max\{R(m_i) + f_{i-1}[x_i - c(m_i)]\}$, $i = 2, 3, \dots, n$.

for possible m_i,when n is the total number of stages,R (m_i) is the measure of performance (like: Return) due to alternative m_i at stage i, $c(m_i)$ is the

cost/resource required for the alternative m_i of the stage i and $f_i(x_i)$ is the value of the measure of performance up to the current stage i from stage 1, if the amount of resource allocated up to the current stage is x_i when forward recursion is used.

Best recursive function value: In a given stage i, the lowest (minimization problem)/ highest (maximization problem) value of the recursive function for a given value of $x_i$is known as the best recursive function value.

Best alternative in a given stage i: In a given stage i , the alternative corresponding to the best recursive function value for a given value of x_i is known as the best alternative for that value of x_i.

Backward recursive function: here computation begins from the last stage/ sub problem, and this stage will be numbered at stage 1, while the first sub problem will be numbered as the last stage. This type of recursive function is known as backward recursive function.

Forward recursive function: While defining the stages of he original problem, the first sub problem will be numbered as stage 1 2 and the last sub problem will be numbered as the last stage. Then, the recursive function will be defined as per this assumption. This type of recursive function is known as forward recursive function.

3. Dynamic programming algorithm:

The computation procedure for solving a problem by dynamic programming approach can be summarized in the following ways.

a) Identify the decision variables and specify objective function to be optimized under certain limitations, if any.
b) Decompose the given problem in to a number of smaller sub problems (or stages).
c) Identify the stage variables at each stage and write down the transformation function as a function of the state variables and decision variables at the next stage.

d) Write down a general recursive relationship for computing the optimal policy. Decide whether forward or backward method is to follow to solve the problem.

e) Construct appropriate stages to show the required values of the return function at each stage.

f) Determine the overall optimal policy or decisions and its value at each stage. There may be more than one such optimal policy.

4. Applications of Dynamic programming.

The dynamic programming can be applied to many real life situations. Some of them are given below:

a) Capital budgeting problem

b) Reliability improvement problem

c) Stage-catch problem (Shortest path problem)

d) Cargo loading problem

e) Minimizing total tardiness in single machine scheduling problem

f) Optimal sub division problem

g) Linear Programming Problem

Example 1. Cargo Loading Problem:

Bita Novelty Company has to load as cargo out of three items whose details are shown below. The maximum weight of the cargo is 5 tons. Find the optimal cargo loading using dynamic programming such that the total return is maximized.

Item(i):	1	2	3
Weight w_i(in Tons):	2	1	4
Return r_i(inRs.):	400	800	200

Solution: Inthis problem, each item i treated as a stage starting from stage 1 to stage 3 for the item 1 to 3, respectively. The maximum weight of the Cargo is 5 tons so, the weight allocated to the alternative in each of the stages are zero and the multiplier of the unit weight (less than or equal to 5) of the item corresponding to that stage.

In stage 1, there are three alternatives and the weights allocated to those alternatives are 0, 2, and 4. The range of values of the state variable in each stage is from 0 to 5(the maximum weight of the Cargo) with increment of 1.

Stage 1.

The recursive function of this stage is

$$\mathrm{F1(x1)} = \frac{\text{Allocated weight}}{\text{Weight of item 1}} \times \text{Returnof item 1}$$

$$= \frac{Allocated\ weight}{2} \times 400$$

Table 1: Calculation of $f_1(x_1)$ for stage 1.

	Alternative m_1			Maxi $f_1(x_1)$ = f_1^*	Best alternative No. m_1^*
State Variable x_1	1	2	3		
	Allocated weight(multiple of 2 less than or equal to 5) 0 2 4				
0	0	----	----	0	1
1	0	----	----	0	1
2	0	400	----	400	2
3	0	400	----	400	2
4	0	400	800	800	3
5	0	400	800	800	3

In Table 1 the calculation of recursive function $f_1(x_1)$ is shown. The Column 1 denote the value of state

variable x_1 which is from 0 to the capacity of the Cargo(here it is 5). Next three columns denote the values of $f_1(x_1)$ based on the three alternatives for allocation of the weight. For item 1 the weight is 2 tones, so possible allocation of weight will be in multiple of 2, and not more then 5, i.e. 0 , 2 and 4. The last column gives the value of maximum $f_1(x_1)$ and corresponding best alternative, denoted by m_1^*.

The calculation of column 2. We put allocation weight 0 in the expression of $f_1(x_1)$ we get 0 answer for x_1. Similarly for allocation weight 2, we put 2 in place of allocation weight in $f_1(x_1)$ we get the answer 400. Here we put 400 for those x_1 which are at least equal to allocated weight and finally for allocation weight 4 the answer for the values becomes 800.

Now the maximum value of $f_1(x_1)$ for $x_1 = 0$ and 1 are 0 and it is for alternative 1 so we put 1 for m_1^* for both the cases.

For $x_1 = 2$, and 3 the maximum value is 400 , which is for alternative 2, so $f_1(x_1) = 400$ and $m_1^* = 2$.

Similarly for $x_1 = 4$, and 5 the maximum value is

800 and corresponding alternative is 3 hence $f_1(x_1)$ = 800 andm_1^* = 3.

Now we move to stage 2.

Stage 2.

The Recursive function of this stage can be obtained as follow.

$$f_2(x_2) = \frac{\text{Allocated weight}}{\text{Weight of item 2}} \times \text{Return on item 2} + f_1(x_2 - \text{Allocated weight})$$

$$= \frac{\text{Allocated weight}}{1} \times 800 + f_1(x_2 - \text{Allocated weight})$$

In stage 2, the weight allocated to the alternatives are zero and the multiplier of the weight 1 of item 2(less than or equal to 5) then there are six alternatives and weights are allocated to those alternatives are 0, 1, 2, 3, 4, and 5.

Table 2. Calculations for stage 2

	Alternative m_2							
State	1	2	3	4	5	6		
Variable x_2	Allocated weight(multiple of 1 less than or equal to 5) 0 1 2 3 4 5						f_2^*	m_2^*

0	0	----	----	----	----	----	0	1
1	0	800	----	----	----	----	800	2
2	400	800	1600	----	----	----	1600	3
3	400	1200	1600	2400	----	----	2400	4
4	400	1200	2000	2400	3200	----	3200	5
5	400	1600	2000	2800	3200	4000	4000	6

Here we have 6 alternatives with possible allocation weight 0, 1, 2 ,3, 4, and 5. The calculations are made in similar way of the calculations in the Table 1. The values of f_2^* = max$f_2(x_2)$ and m_2^* are listed in the last two columns.

Stage 3.

The Recursive function of this stage can be obtained as follow.

$$f_3(x_3) = \frac{\text{Allocated weight}}{\text{Weight of item 3}} \times \text{Return on item 3}$$

$$+ \ f_2(x_3 - \text{ Allocated weight})$$

$$= \frac{\text{Allocated weight}}{4} \times 200 + \ f_2(x_3 - \text{ Allocated weight})$$

In stage 3, the weight allocated to the alternative are zero and the multiplier of the weight 4 of item 2 (less than or equal to 5) so there are two alternatives and the weight allocated to these alternatives are 0 and 4.

Table 3: Calculation for stage 3.

State Variable x_3	Alternative m_3		f_3^*	m_3^*
	1	2		
	Allocated weight 0 4			
0	0	---	0	1
1	800	---	800	1
2	1600	---	1600	1
3	2400	---	2400	1
4	3200	200	3200	1

5	4000	1000	4000	1

Tracing the solution:

In the last two columns the maximum value of $f_3(x_3)$ is 4000 for alternative 1 and its allocated weight is 0, i.e $x_3 = 0$

Now for second unit x_2 = capacity – allocated weight for x_3

= 5-0 = 5

Therefore $x_2 = 5$. Now fromTable 2, for $x_2 = 5$ the $m_2 = 6$, for which allocated weight is 5.

therefore for first unit x_1 = capacity – allocated weight for x_2

= 5 –5 = 0.

Summary of weight of items in the Cargo

item:	1	2	3
weight(in tons):	0	5	0
return:	400	800	200

Total return: $0 \times 400 + 5 \times 800 + 0 \times 200 = 4000$

5. Solution of linear programming problem (LPP) through Dynamic programming:

Let us consider the generalized linear programming problem as

Max $Z = c_1x_1 + c_2x_2 + \ldots + c_jx_j + \ldots + c_nx_n$

subject to the conditions

$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n \leq b_1$

$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n \leq b_i$

$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n \leq b_m$

$x_j \geq 0, j = 1, 2, \ldots, n.$

In this lpp the project j is treated as j, when j = 1, 2, … ,n so we consider the total number of stages of the problem = n. At stage j, a value of the decision variable X_j is known as alternative.

Here X_j is a continuous variable, there will be infinite number of alternatives in each stage j = 1,2, ... ,n.

Here we use the method of back recursion to solve the problem.

The project 1 is treated as stage 1

The project 2 is treated as stage 2 and so on.

Let us suppose that b_{ij} = state of the system with respect to constraint i and $f_j(b_{1j}, b_{2j}, \ldots b_{ij}, \ldots, b_{mj})$ = the optimum objective function value at stage j.

Now the objective function for stage n is

$f_n(b_{1n}, b_{2n}, \dots b_{in}, \dots, b_{mn}) = \max(c_n x_n)$, $0 \le a_{in}x_n \le b_{in}$, for i = 1, 2, … ,m.

And the objective function for stage j is

$f_j(b_{1j}, b_{2j}, \dots b_{ij}, \dots, b_{mj}) = \max(c_j x_j + f_{j+1}[(b_{1j} - a_{1j}x_j),(b_{2j} - a_{2j}x_j),$

$\dots\dots(b_{mj} - a_{mj}x_j)])$,

over $0 \le a_{ij}x_j \le b_{ij}$; $0 \le bij \le b_i$ for i = 1, 2, … ,m, j= 1,2, …, n.

Let us consider an example to understand the method.

Example 2:

Solve the LPP using dynamic programming.

Max $Z = 30x_1 + 40x_2$,

subject to $3x_1 + 6x_2 \le 210$, $8x_2 \le 240$, x_1 and $x_2 \ge 0$.

Solution: Here the number of decision variables is two so, we have two stages. Stage 1 for variable X_1 and stage 2 for x_2. Since backward recursion is used to solve the problem.

Stage 2 is considered first.

The set of states of different stages are given in the following Table 4

Table 4: Set of states of different stages

State j	Decision Variable	Set of States
2	X_2	$\{ b_{12}, b_{22}\}$
1	X_1	$\{b_{11}, b_{21}\}$

Based on the backward recursive function for stage 2 with respect to x_2 is given by

$f_2(b_{12}, b_{22}) = \max 40x_2$ subject to $0 \leq 6x_2 \leq b_{12}$, $0 \leq 8x_2 \leq b_{22}$

To maintain feasibility x_2 should be the mini$\{ b_{12}/6, b_{22}/8 \}$.

Hence the above objective function reduces to

$f_2(b_{12}, b_{22}) = 40 \min (b_{12}/6, b_{22}/8)$

and$x^*_2 = \min \{ b_{12}/6, b_{22}/8 \}$.

..(1)

Now recursive function for stage 1 with respect to x_1 is

$f_1(b_{11}, b_{21}) = \max[30x_1 + f_2(b_{11} -3x_1, b_{21})]$, $0 \leq 3x_1 \leq b_{11}$ ..(2)

$= \max[30x_1 + 40 \min((b_{11} - 3x_1)/6, b_{21}/8)]$,

$0 \leq 3x_1 \leq b_{11}$

This stage is the last in the series of backward recursion.

As $b_{11} = 210$ and $b_{21} = 240$, so to determine the upper limit for x_1^* we have

$f_1(x_1/b_{11}, b_{21}) = \max[f_1(x_1/210, 240)]$

$= \max[30x_1 + 40\min\{(210-3x_1)/6, 240/8\}]$

Now, to which in the ranges of x_1 is defined, $(210-3x_1)/6$ can be as high as 30 or as low as 0. So, equate it to 30 as well as 0 and solve for x_1 we get

$x_1 = 10$ and $x_1 = 70$

Therefore the range for x_1 is $0 \leq x_1 \leq 10$ and $10 \leq x_1 \leq 70$.

Now $f_1(x_1/b_{11}, b_{21})$ is re written as

$f_1(x_1/b_{11}, b_{21}) = \max[30x_1 + 40\min\{(210-3x_1)/6, 30\}]$, $0 \leq x_1 \leq 10$

$= \max[30x_1 + 40\min\{(210-3x_1)/6, 30\}]$, $10 \leq x_1 \leq 70$

$f_1(x_1/210, 240) = \max[30x_1 + 40 \times 30]$, $0 \leq x_1 \leq 10$

$= \max[30x_1 + 40 \times (210-3x_1)/6]$, $10 \leq x_1 \leq 70$

$f_1(x_1/210, 240)=\max[30x_1 + 1200], 0\leq x_1\leq 10$

$=\max[30x_1 + 1400\text{-}20x_1], 10\leq x_1\leq 70$

To maximize each of the above cases substitute 10 for x_1 we get

$f_1(x_1/210, 240)=\max(1500, 1500) = 1500$

Therefore $X_1^* = 10$ and $f_1(x_1/210, 240)=1500$.

..(3)

For tracing the value of x_2^* we have from (2) and (1)

$b_{12} = b_{11} - 3x_1 = 210 - 3(10) = 210 - 30 = 180$

$b_{22} = b_{21} = 240$

therefore

$x_2^* = \min \{ b_{12}/6, b_{22}/8 \}= \min(180/6, 240/8) = 30$

..(4)

Thus the optimum solution from (3) and (4) is

$X_1 = 10$, $x_2 = 30$ and max $Z = 1500$.

Summary:

In Operations research there are many methods of solving decision problems. Dynamic programming is a method of solving a problem based on the theory of multistage decision process. There is a sequence of interrelated decisions has to be made at in each stage. In Dynamic programming the original

problem is sub-divided into n sub-problems. Each sub-problem is a decision –making problem in one variable only. The solution of the original problem is achieved by integrating the solutions of the sub-problems. Thus it is a systematic, complete enumeration technique. Many types of decision problems can be solved by dynamic programming.

Glossary:

Stage i:

Each sub problem of the original problem is known as stage i.

Alternative m_i:

In a given stage i, there may be more than one choice of carrying out a test. Each choice is known as alternative m_i.

State variable x_i:

A possible value of resource within its permitted range at a given stage I is known as state variable x_i.

Recursive function $f_i(x_i)$:

A function which links the measure of performance of interest of the current stage with the cumulative measure of performance of the previous stages/ succeeding stages as a function of the state variable of the current stage is known as the recurrence function of the current stage.

Best recursive function value:

In a given stage i, the lowest (minimization problem)/ highest (maximization problem) value of the recursive function for a given value of x_i known as the best recursive function value.

Best alternative in a given stage i:

In a given stage i , the alternative corresponding to the best recursive function value for a given value of x_i is known on the best alternative for that value of x_i.

Backward recursive function:

Here computation begins from the last stage/ sub problem, and this stage will be numbered at stage 1, while the first sub problem will be numbered as the last stage. This type of recursive function is known as backward recursive function.

Forward recursive function:

While defining the stages of he original problem, the first sub problem will be numbered as stage 1 and last sub problem will be numbered as the last stage. Then , the recursive function will be defined as per this assumption. This type of recursive function is known as forward recursive function.