Co-funded by the
Erasmus+ Programme
of the European Union

Advanced Optimization: Techniques and Industrial Applications

Universidade do Minho

Curriculum Development
of Master's Degree Program in
Industrial Engineering for Thailand Sustainable Smart Industry

Co-funded by the Erasmus+ Programme of the European Union

Session 1.5: Dynamic Programming

Universidade do Minho

Curriculum Development of Master's Degree Program in Industrial Engineering for Thailand Sustainable Smart Industry

Introduction

- A specific type of mathematical programming in which the optimal solution of the original problem is found by solving *a chain of subproblems*.

- In dynamic programming, the optimal solution of one subproblem will be used as input to the next subproblem. When the last subproblem is solved, the optimal solution for the *entire* problem is achieved, which includes the solution of the original problem.

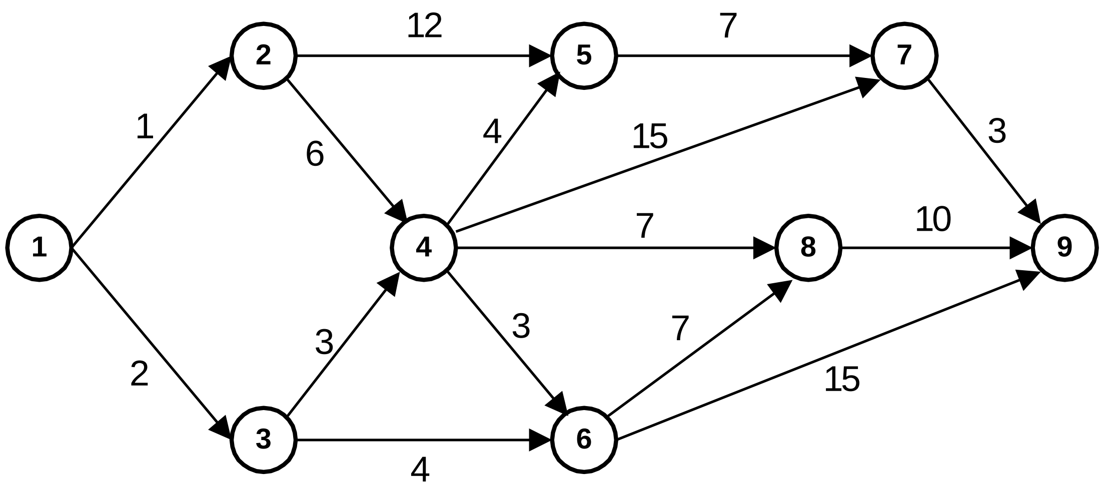
- Linkage between the stages of a DP problem is performed through *recursive computations*. Depending on the nature of the problem at hand, *forward recursive equation* or *backward recursive equation* will be developed for finding solution.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

<u>Example 1</u>: Shortage Path Problem

Find the shortage path from node 1 to node 9 of the network:



Define:

f_i: minimum total travel time from node *i* to node 9.

t_{ij} : travel time through the directed arc (i, j).

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

On an arbitrary arc (i, j), it can be seen that:

$$f_i \leq t_{ij} + f_j \qquad i \neq 9, \forall j$$

Hence:

$$f_i \leq \min_j \{t_{ij} + f_j\} \qquad i \neq 9$$

However, the shortage path from node *i* to node 9 should include some intermediate node *j* (if these intermediate nodes exist). Then,

$$f_i = \min_j \{t_{ij} + f_j\} \qquad i \neq 9$$

The above equation is the *recursive equation* (or functional equation) of the shortage path problem in the *backward* form.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

Based on the recursive equation, the optimal solution can be found as follows:

$$f_9 = 0$$

$$f_8 = t_{89} + f_9 = 10 + 0 = 10$$

$$f_7 = t_{79} + f_9 = 3 + 0 = 3$$

$$f_6 = \min\begin{Bmatrix} t_{68} + f_8 \\ t_{69} + f_9 \end{Bmatrix} = \min\begin{Bmatrix} 7 + 10 \\ 15 + 0 \end{Bmatrix} = 15$$

$$f_5 = t_{57} + f_7 = 7 + 3 = 10$$

$$f_4 = \min\begin{Bmatrix} t_{45} + f_5 \\ t_{46} + f_6 \\ t_{47} + f_7 \\ t_{48} + f_8 \end{Bmatrix} = \min\begin{Bmatrix} 4 + 10 \\ 3 + 15 \\ 15 + 3 \\ 7 + 10 \end{Bmatrix} = 14$$

Co-funded by the Erasmus+ Programme of the European Union

Introduction

$$f_3 = \min\begin{Bmatrix} t_{34} + f_4 \\ t_{36} + f_6 \end{Bmatrix} = \min\begin{Bmatrix} 3 + 14 \\ 4 + 15 \end{Bmatrix} = 17$$

$$f_2 = \min\begin{Bmatrix} t_{24} + f_4 \\ t_{25} + f_5 \end{Bmatrix} = \min\begin{Bmatrix} 6 + 14 \\ 12 + 10 \end{Bmatrix} = 20$$

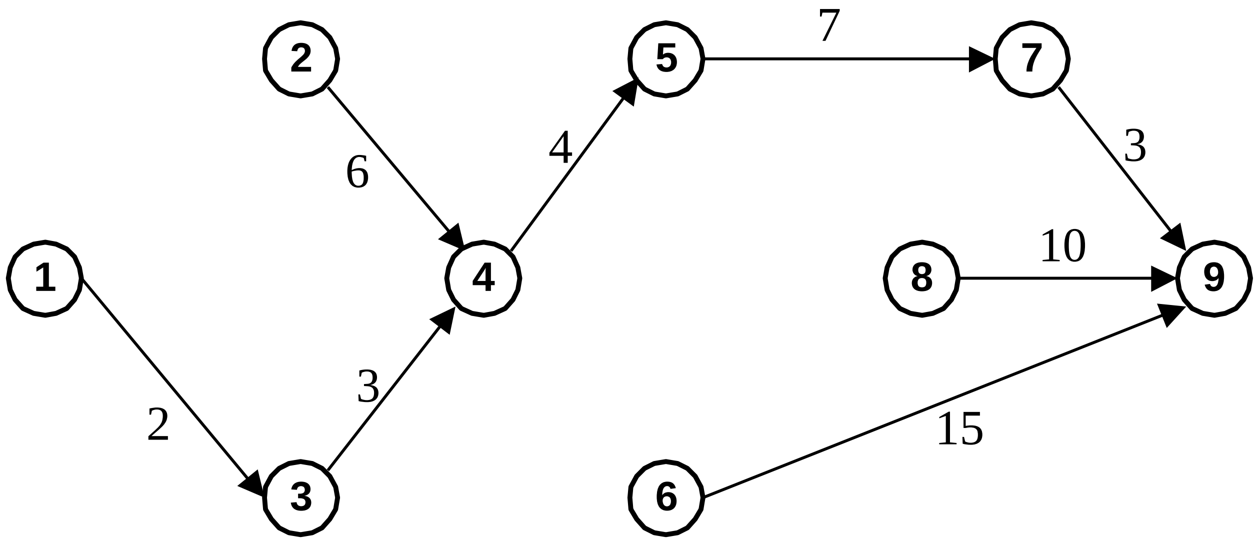
$$f_1 = \min\begin{Bmatrix} t_{12} + f_2 \\ t_{13} + f_3 \end{Bmatrix} = \min\begin{Bmatrix} 1 + 20 \\ 2 + 17 \end{Bmatrix} = 19$$

Shortage path from node 1 to node 9: 1-3-4-5-7-9 with total travel time = 19.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

It is noted that each subproblem is associated with a network’s node and when the shortage path from node 1 to node 9 is determined, we also know the shortage paths from every nodes of the network to node 9.



Co-funded by the Erasmus+ Programme of the European Union

Introduction

The above problem can also be solved by use of *forward* recursive equation as presented below

Define:

f_j: minimum total travel time from node 1 to node *j*.

t_{ij} : travel time through the directed arc (i, j).

On an arbitrary arc (i, j), it can be seen that:

$$f_j \leq t_{ij} + f_i \qquad j \neq 1, \forall i$$

Hence:

$$f_j \leq \min_i \{t_{ij} + f_i\} \qquad j \neq 1$$

However, the shortage path from node 1 to node *j* should include some intermediate node *i* (if these intermediate nodes exist). Then,

$$f_j = \min_i \{t_{ij} + f_i\} \qquad j \neq 1$$

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

Solution can be found recursively as follows:

$$f_1 = 0$$

$$f_2 = t_{12} + f_1 = 1 + 0 = 1$$

$$f_3 = t_{13} + f_1 = 2 + 0 = 2$$

$$f_4 = \min \begin{Bmatrix} t_{24} + f_2 \\ t_{34} + f_3 \end{Bmatrix} = \min \begin{Bmatrix} 6 + 1 \\ 3 + 2 \end{Bmatrix} = 5$$

$$f_5 = \min \begin{Bmatrix} t_{25} + f_2 \\ t_{45} + f_4 \end{Bmatrix} = \min \begin{Bmatrix} 12 + 1 \\ 4 + 5 \end{Bmatrix} = 9$$

$$f_6 = \min \begin{Bmatrix} t_{36} + f_3 \\ t_{64} + f_4 \end{Bmatrix} = \min \begin{Bmatrix} 4 + 2 \\ 3 + 5 \end{Bmatrix} = 6$$

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

$$f_7 = \min\begin{Bmatrix} t_{47} + f_4 \\ t_{57} + f_5 \end{Bmatrix} = \min\begin{Bmatrix} 15 + 5 \\ 7 + 9 \end{Bmatrix} = 16$$

$$f_8 = \min\begin{Bmatrix} t_{48} + f_4 \\ t_{68} + f_6 \end{Bmatrix} = \min\begin{Bmatrix} 7 + 5 \\ 7 + 6 \end{Bmatrix} = 12$$

$$f_9 = \min\begin{Bmatrix} t_{69} + f_6 \\ t_{79} + f_7 \\ t_{89} + f_8 \end{Bmatrix} = \min\begin{Bmatrix} 15 + 6 \\ 3 + 16 \\ 10 + 12 \end{Bmatrix} = 19$$

The solution from forward recursive equation gives the shortage paths from node 1 to every other nodes of the network, not only to node 9.

Co-funded by the Erasmus+ Programme of the European Union

Introduction

Note:

Not all DP programs can be solved by both forward and backward recursive techniques. The use of backward recursion or forward recursion depends on the specific structure of the problem under consideration.

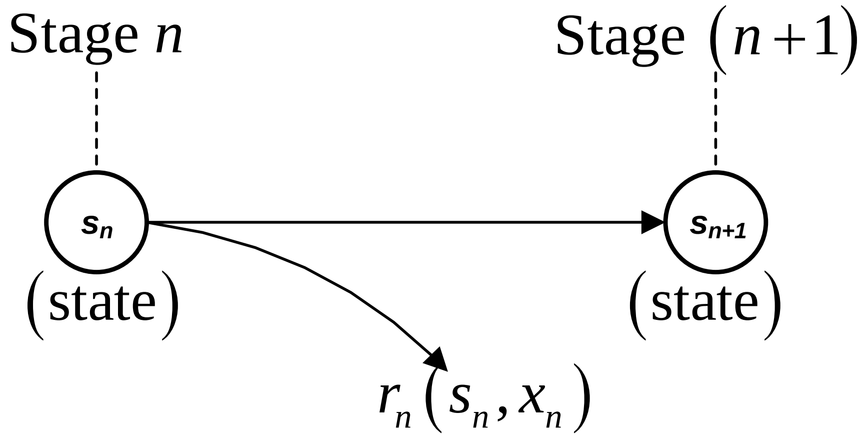
Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

Bellman's Principle of Optimality

An optimal policy has the property that whatever the initial state and the initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision

The basic DP approach can be illustrated by the following diagram:



Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

- At state s_n in stage n, if decision x_n is taken the current state s_n will be transferred to a new state s_{n+1} in stage $(n+1)$.

- A revenue $r_n(s_n, x_n)$ will be obtained by decision x_n taken at state s_n

- The new state s_{n+1} is also a function of s_n and x_n, and can be expressed in form of a transformation function: $s_{n+1} = t_n(s_n, x_n)$.

Co-funded by the Erasmus+ Programme of the European Union

Introduction

Bellman's Principle of Optimality

In case of maximization problem and backward recursive technique is employed, if we denote $f_n(s_n)$ as the maximum total revenue obtained when the system moves from stage n to stage N (the last stage), given the observed state at stage n is s_n, then:

$$f_n(s_n) = \max_{x_n \in D(s_n)} \left\{ r_n(s_n, x_n) + f_{n+1}\left(t_n(s_n, x_n)\right) \right\}$$

In which $D(s_n)$ is the set of all possible decisions of a *given* state s_n at stage n (decision set).

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

Similarly, when the forward recursive technique is employed, if we denote $f_n(s_n)$ as the maximum total revenue when the system move from stage 1 to stage n, given the observed state at stage n is s_n, then:

$$f_{n+1}\left(s_{n+1}\right) = \max_{x_n \in D\left(s_{n+1}\right)} \left\{ r_n\left(s_n, x_n\right) + f_n\left(s_n\right) \right\}$$

In which $D(s_{n+1})$ is the set of all possible decisions x_n at stage n such that these decisions will help to transfer the states s_n's at stage n to a predefined state s_{n+1} in stage $(n+1)$, i.e., $s_{n+1} = t_n(s_n, x_n)$.

Co-funded by the
Erasmus+ Programme
of the European Union

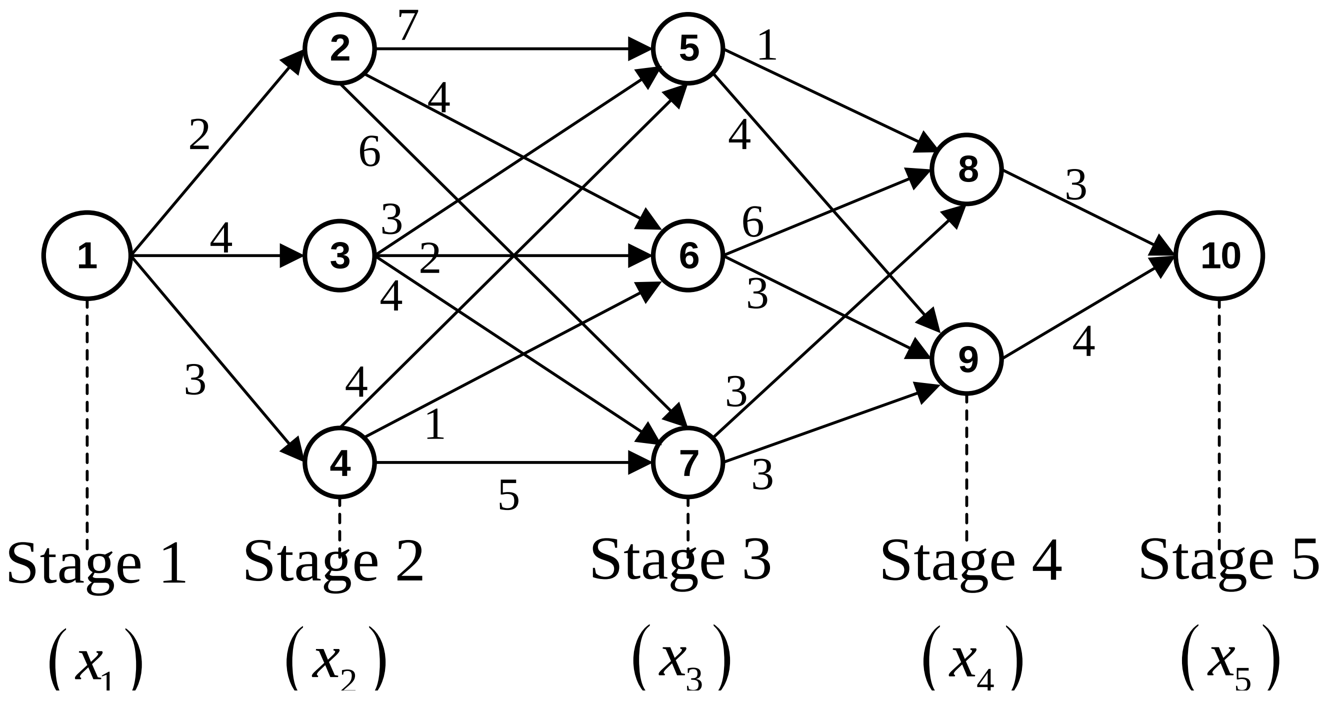
Introduction
Bellman’s Principle of Optimality

The Bellman’ optimality principle can help to establish recursive equations when the structure of the problem can be arranged in stages.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

Example 2: Find the shortage path from node 1 to node 10 of the network



Co-funded by the Erasmus+ Programme of the European Union

Introduction
Bellman's Principle of Optimality

Applying backward recursive approach, the solution can be found as follows:

<u>Stage 4</u>:

s_4 \ x_4	$r_4(s_4,x_4)+f_5(t_4(s_4,x_4))$: 10	$f_4(s_4)$	x_4^*
Node 8	3	3	10
Node 9	4	4	10

In stage 4, the state s_4 can be *node* 8 or *node* 9, and the only decision (i.e., x_4) that can be taken is *go to node* 10.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

When the state is *node* 8:

$$r_4(s_4, x_4) = r_4(8, 10) = 3 \qquad f_5\big(t_4(s_4, x_4)\big) = f_5(10) = 0$$

$$\Rightarrow \qquad f_4(s_4) = f_4(8) = 3$$

When the state is *node* 9:

$$r_4(s_4, x_4) = r_4(9, 10) = 4 \qquad f_5\big(t_4(s_4, x_4)\big) = f_5(10) = 0$$

$$\Rightarrow \qquad f_4(s_4) = f_4(9) = 4$$

Due to the fact that there exists only one possible decision, that decision is the optimal decision.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

Stage 3:

x_3 / s_3	$r_3(s_3,x_3)+f_4(t_3(s_3,x_3))$		$f_3(s_3)$	x_3^*
	8	9		
Node 5	1 + 3 = 4	4 + 4 = 8	4	8
Node 6	6 + 3 = 9	3 + 4 = 7	7	9
Node 7	3 + 3 = 6	3 + 4 = 7	6	8

In this stage, the state s_3 can be *node* 5, *node* 6 or *node* 7. The decision x_3 can be *go to node* 8 or *go to node* 9.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

When the state is *node* 5:

If the decision is *go to node* 8:

$$r_3(s_3, x_3) = r_3(5,8) = 1 \qquad f_4\big(t_3(s_3, x_3)\big) = f_4(8) = 3$$

If the decision is *go to node* 9:

$$r_3(s_3, x_3) = r_3(5,9) = 4 \qquad f_4\big(t_3(s_3, x_3)\big) = f_4(9) = 4$$

$\Rightarrow \qquad f_3(s_3) = f_3(5) = \mathrm{Min}\{1 + 3, 4 + 4\} = 4$

$\Rightarrow$ Optimal decision: *go to node* 8

Co-funded by the Erasmus+ Programme of the European Union

Introduction
Bellman's Principle of Optimality

When the state is *node* 6:

If the decision is *go to node* 8:

$$r_3(s_3, x_3) = r_3(6, 8) = 6 \qquad f_4\big(t_3(s_3, x_3)\big) = f_4(8) = 3$$

If the decision is *go to node* 9:

$$r_3(s_3, x_3) = r_3(6, 9) = 3 \qquad f_4\big(t_3(s_3, x_3)\big) = f_4(9) = 4$$

$\Rightarrow \qquad f_3(s_3) = f_3(6) = \text{Min}\{6 + 3, 3 + 4\} = 7$

$\Rightarrow$ Optimal decision: *go to node* 9

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

When the state is *node 7*:

If the decision is *go to node* 8:

$$r_3(s_3, x_3) = r_3(7,8) = 3 \qquad f_4\big(t_3(s_3, x_3)\big) = f_4(8) = 3$$

If the decision is *go to node* 9:

$$r_3(s_3, x_3) = r_3(7,9) = 3 \qquad f_4\big(t_3(s_3, x_3)\big) = f_4(9) = 4$$

$\Rightarrow \qquad f_3(s_3) = f_3(7) = \text{Min}\{3+3, 3+4\} = 6$

$\Rightarrow$ Optimal decision: *go to node* 8

Co-funded by the
Erasmus+ Programme
of the European Union

Stage 2:

s_2 \ x_2	$r_2(s_2,x_2)+f_3(t_2(s_2,x_2))$			Min $f_2(s_2)$	x_2^*
	5	6	7		
Node 2	7 + 4 = 11	4 + 7 = 11	6 + 6 = 12	11	5 or 6
Node 3	3 + 4 = 7	2 + 7 = 9	4 + 6 = 10	7	5
Node 4	4 + 4 = 8	1 + 7 = 8	5 + 6 = 11	8	5 or 6

In this stage, the state s_2 can be *node* 2, *node* 3 or *node* 4. The decision x_2 can be *go to node* 5, *go to node* 6, or *go to node* 7.

Co-funded by the Erasmus+ Programme of the European Union

Bellman's Principle of Optimality

When the state is *node 2*:

If the decision is *go to node* 5:

$$r_2(s_2, x_2) = r_2(2,5) = 7 \qquad f_3\big(t_2(s_2, x_2)\big) = f_3(5) = 4$$

If the decision is *go to node* 6:

$$r_2(s_2, x_2) = r_2(2,6) = 4 \qquad f_3\big(t_2(s_2, x_2)\big) = f_3(6) = 7$$

If the decision is *go to node 7*:

$$r_2(s_2, x_2) = r_2(2,7) = 6 \qquad f_3\big(t_2(s_2, x_2)\big) = f_3(7) = 6$$

$\Rightarrow$ $f_2(s_2) = f_2(2) = \text{Min}\{7 + 4, 4 + 7, 6 + 6\} = 11$

$\Rightarrow$ Optimal decision: *go to node* 5 or *go to node* 6

Co-funded by the
Erasmus+ Programme
of the European Union

Bellman's Principle of Optimality

When the state is *node* 3:

If the decision is *go to node* 5:

$$r_2(s_2, x_2) = r_2(3,5) = 3 \qquad f_3\big(t_2(s_2, x_2)\big) = f_3(5) = 4$$

If the decision is *go to node* 6:

$$r_2(s_2, x_2) = r_2(3,6) = 2 \qquad f_3\big(t_2(s_2, x_2)\big) = f_3(6) = 7$$

If the decision is *go to node 7*:

$$r_2(s_2, x_2) = r_2(3,7) = 4 \qquad f_3\big(t_2(s_2, x_2)\big) = f_3(7) = 6$$

$$\Rightarrow \qquad f_2(s_2) = f_2(3) = \text{Min}\{3+4, 2+7, 4+6\} = 7$$

$\Rightarrow$ Optimal decision: *go to node* 5

Co-funded by the
Erasmus+ Programme
of the European Union

Bellman's Principle of Optimality

When the state is *node 4*:

If the decision is *go to node 5*:

$r_2(s_2, x_2) = r_2(4,5) = 4$ $\quad\quad$ $f_3(t_2(s_2, x_2)) = f_3(5) = 4$

If the decision is *go to node 6*:

$r_2(s_2, x_2) = r_2(4,6) = 1$ $\quad\quad$ $f_3(t_2(s_2, x_2)) = f_3(6) = 7$

If the decision is *go to node 7*:

$r_2(s_2, x_2) = r_2(4,7) = 5$ $\quad\quad$ $f_3(t_2(s_2, x_2)) = f_3(7) = 6$

$\Rightarrow$ $\quad f_2(s_2) = f_2(3) = \text{Min}\{4+4, 1+7, 5+6\} = 8$

$\Rightarrow$ $\quad$ Optimal decision: *go to node 5* or *go to node 6*

Co-funded by the
Erasmus+ Programme
of the European Union

Stage 1:

s_1 \ x_1	$r_1(s_1,x_1)+f_2(t_1(s_1,x_1))$			$f_1(s_1)$	x_1^*
	2	3	4		
Node 1	2 + 11 = 13	4 + 7 = 11	3 + 8 = 11	11	3 or 4

In this stage, the state s_1 is *node* 1. The decisions x_1 can be *go to node* 2, *go to node* 3, or *go to node* 4.

Co-funded by the Erasmus+ Programme of the European Union

Bellman's Principle of Optimality

If the decision is *go to node 2*:

$$r_1(s_1, x_1) = r_1(1,2) = 2 \qquad f_2\big(t_1(s_1, x_1)\big) = f_2(2) = 11$$

If the decision is *go to node 3*:

$$r_1(s_1, x_1) = r_1(1,3) = 4 \qquad f_2\big(t_1(s_1, x_1)\big) = f_2(3) = 7$$

If the decision is *go to node 7*:

$$r_1(s_1, x_1) = r_1(1,4) = 3 \qquad f_2\big(t_1(s_1, x_1)\big) = f_2(4) = 8$$

$\Rightarrow \qquad f_1(s_1) = f_1(1) = \mathrm{Min}\{2+11, 4+7, 3+8\} = 11$

$\Rightarrow$ Optimal decision: *go to node 3* or *go to node 4*

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman’s Principle of Optimality

The optimal solution has been found. There exist three shortage paths from node 1 to node 10:

1-3-5-8-10, 1-4-5-8-10, 1-4-6-9-10

Co-funded by the Erasmus+ Programme of the European Union

Example 3:

Five medical teams will be dispatched to 3 regions to help improve medical care. The performance is measured by the expected additional person-years of life. The estimated performance measures are given in the table:

No. of Teams	Additional person-years life (in 1000 units)		
	Region 1	Region 2	Region 3
0	0	0	0
1	45	20	50
2	70	45	70
3	90	75	80
4	105	110	100
5	120	150	130

Co-funded by the Erasmus+ Programme of the European Union

Introduction
Bellman's Principle of Optimality

The problem is to allocate the medical teams so that the total additional person-years of life can be maximized.

Denote:

x_n : number of teams to be allocated to region *n* (*n* = 1, 2, 3).

s_n : number of teams available for allocation to the regions *n*, *n* +1,…, 3.

$p_n(x_n)$: the measure of performance from allocation x_n teams to region *n*.

$f_n(s_n)$: Total maximum performance obtained when s_n teams are allocated to regions *n*, *n* +1,…, 3.

Co-funded by the Erasmus+ Programme of the European Union

Introduction
Bellman's Principle of Optimality

The problem can be formulated as follows:

$$f_1(5) = \text{Max}\, p_1(x_1) + p_2(x_2) + p_3(x_3)$$

s.t. $\quad x_1 + x_2 + x_3 \leq 5$

$\quad x_j \geq 0$ and integer $\forall j = 1,2,3$

The above problem can be considered as embedded in the following chains of subproblems:

$$f_n(s_n) = \text{Max}\ \sum_{i=n}^{3} p_i(x_i)$$

s.t. $\quad \sum_{i=n}^{3} x_i \leq s_n$

$\quad x_j \geq 0$ and integer $\forall j = n, \ldots, 3$

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

The backward recursive equation can be developed as:

$$f_n(s_n) = \max_{\substack{0 \le x_n \le s_n \\ x_n \text{ integer}}} \{ p_n(x_n) + f_{n+1}(s_n - x_n) \}$$

Noting that $f_4(s_4) = 0$, the solution can be obtained as follows:

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction

Bellman's Principle of Optimality

$\underline{n = 3}$

No. of teams	$p_3(x_3)+f_4(s_4)=p_3(x_3)+f_4(s_3-x_3)$						Max $f_3(s_3)$	x_3^*
	$x_3=0$	$x_3=1$	$x_3=2$	$x_3=3$	$x_3=4$	$x_3=5$		
$s_3=0$	0	-	-	-	-	-	0	0
$s_3=1$	0	**50**	-	-	-	-	**50**	1
$s_3=2$	0	50	70	-	-	-	70	2
$s_3=3$	0	50	70	80	-	-	80	3
$s_3=4$	0	50	70	80	100	-	100	4
$s_3=5$	0	50	70	80	100	130	130	5

Co-funded by the Erasmus+ Programme of the European Union

Introduction

Bellman's Principle of Optimality

$n = 2$

No. of teams	$x_2 = 0$	$x_2 = 1$	$x_2 = 2$	$x_2 = 3$	$x_2 = 4$	$x_2 = 5$	Max $f_2(s_2)$	x_2^*
	$p_2(x_2) + f_3(s_3) = p_2(x_2) + f_3(s_2 - x_2)$							
$s_2 = 0$	0+0 = 0	-	-	-	-	-	0	0
$s_2 = 1$	0+50 = 50	20+0 = 20	-	-	-	-	50	0
$s_2 = 2$	0+70 = 70	20+50 = 70	45+0 = 45	-	-	-	70	0-1
$s_2 = 3$	0+80 = 80	20+70 = 90	45+50 = 95	75+0 = 75	-	-	95	2
$s_2 = 4$	0+100 = 100	20+80 = 100	45+70 = 115	**75+50 = 125**	110+0 = 110	-	**125**	3
$s_2 = 5$	0+130 = 130	20+100 = 120	45+80 = 125	75+70 = 145	110+50 = 160	150+0 = 150	160	4

Co-funded by the Erasmus+ Programme of the European Union

Introduction
Bellman's Principle of Optimality

$n = 1$

No. of teams	$p_1(x_1) + f_2(s_2) = p_1(x_1) + f_2(s_1 - x_1)$						Max $f_1(s_1)$	x_1^*
	$x_1 = 0$	$x_1 = 1$	$x_1 = 2$	$x_1 = 3$	$x_1 = 4$	$x_1 = 5$		
$s_1 = 0$	*	-	-	-	-	-	*	*
$s_1 = 1$	*	*	-	-	-	-	*	*
$s_1 = 2$	*	*	*	-	-	-	*	*
$s_1 = 3$	*	*	*	*	-	-	*	*
$s_1 = 4$	*	*	*	*	*	-	*	*
$s_1 = 5$	0+160 = 160	**45+125 = 170**	70+95 = 165	90+70 = 160	115+50 = 165	120+0 = 120	**170**	1

(*: no need to determine those values)

Optimal solution: $x_1^* = 1, x_2^* = 3, x_3^* = 1$; optimal objective function 170.

Co-funded by the Erasmus+ Programme of the European Union

Example 4: Resource Allocation Problem

Consider the single resource allocation problem to produce *N* products:

$$\begin{aligned} \text{Max} \quad & p_1(x_1) + p_2(x_2) + \cdots + p_N(x_N) \\ \text{s.t.} \quad & c_1(x_1) + c_2(x_2) + \cdots + c_N(x_N) \leq K \\ & x_j \in \Omega_j \qquad \forall j = 1,2,\ldots,N \end{aligned}$$

In which:

$p_j(x_j)$: profit obtained by producing x_j units of product *j*.

$c_j(x_j)$: units of the resource consumed for producing x_j units of product *j*.

Ω_j : the set of possible production levels for product *j*.

The above problem can be solved by dynamic programming

Co-funded by the Erasmus+ Programme of the European Union

The embedded problem in backward recursive form

Define:

- (n, y): state - *y* units of resource are allocated to produce products from *n* through *N*.
- $f_n(y)$: maximum total profit obtained from products *n* through *N*, when *y* units of resource are allocated to them.
- $f_1(K)$: the optimal value to be determined.

Co-funded by the Erasmus+ Programme of the European Union

Notes:

1. $f_n(y)$ can be expressed as:

$$\begin{aligned} \text{Max} \quad & p_n(x_n) + \cdots + p_N(x_N) \\ \text{s.t.} \quad & c_n(x_n) + \cdots + c_N(x_N) \leq y \\ & x_j \in \Omega_j \qquad \forall j = n, \ldots, N \end{aligned}$$

The problem $f_1(K)$ is embedded in the above problems: $f_n(y)$ for $n = 1,2, \ldots, N$ and $y = 0,1,2, \ldots, K$

Co-funded by the Erasmus+ Programme of the European Union

Bellman's Principle of Optimality

2. Boundary conditions:

Max $f_N(y) = \text{Max}\, p_N(x_N)$

s.t. $c_N(x_N) \leq y$

$x_N \in \Omega_N$

Co-funded by the Erasmus+ Programme of the European Union

Introduction
Bellman's Principle of Optimality

Backward recursive equation:

$$f_n(y) = \max_{\substack{c_n(x_n)\le y \\ x_n \in \Omega_n}} \{p_n(x_n) + f_{n+1}(y - c_n(x_n))\}$$

In this case, we have:

$s_n = (n, y)$; $D(s_n) = \{x \in \Omega_n | c_n(x) \le y\}$; and
$s_{n+1} = t_n(s_n, x) = (n+1, y - c_n(x))$

Co-funded by the Erasmus+ Programme of the European Union

Introduction
Bellman's Principle of Optimality

The embedded problem in forward recursive form

Define:

- (n, y): state - *y* units of resource are allocated to produce products from 1 to *n*.
- $f_n(y)$: maximum total profit obtained from products 1 through *n*, when *y* units of resource are allocated to them.
- $f_N(K)$: the optimal value to be determined.

Co-funded by the
Erasmus+ Programme
of the European Union

Introduction
Bellman's Principle of Optimality

Forward recursive equation:

$$f_n(y) = \max_{\substack{c_n(x_n) \le y \\ x_n \in \Omega_n}} \{p_n(x_n) + f_{n-1}(y - c_n(x_n))\}$$

In this case, we still have:

$s_n = (n, y)$; $D(s_n) = \{x \in \Omega_n | c_n(x) \le y\}$; and

$s_{n-1} = t_n(s_n, x) = (n - 1, y - c_n(x))$.

Co-funded by the
Erasmus+ Programme
of the European Union