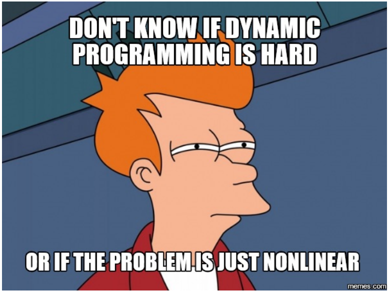
Dynamic Programming III



Class 27

April 24, 2023

EXAMTODAY

Here

7 PM – ?

Dynamic Programming

Dynamic programming is a useful technique that we can use to solve many optimization problems by breaking up large problems into a sequence of smaller, more tractable problems and then **working backward** from the end of the problem toward the beginning of the problem.

Characteristics of Dynamic Programming Problems

Characteristics of Dynamic Programming Problems II

Problem can be divided into stages with a policy decision required at each stage.

- Each stage has a number of states associated with the beginning of the stage.
- The effect of the policy decision at each stage is to transform the current state to a state associated with the beginning of the next stage.
- The solution procedure finds an optimal policy for the overall problem.
- *The Principle of Optimality:* An optimal policy for the remaining stages is independent of the policy decisions adopted at previous stages.
- Solution procedure begins by finding optimal policy for the last stage.
- A recursive relationship that identifies the optimal policy for stage n given the optimal policy for n+1 is known.

$$f_n^*(s) = min[C_{sx_n} + f_{n+1}^*(x_n)]$$

- A recursive relationship that identifies the optimal policy for stage n given the optimal policy for n+1 is known.

$$f_n^*(s) = min[C_{sx_n} + f_{n+1}^*(x_n)]$$

N = number of stages
n = label for current stage ($n = 1, 2, ..., N$)
s_n = current state for stage n
x_n = decision variable for stage n
x_n^* = optimal value for x_n given s_n
$f_n(s_n, x_n)$ = contribution of stages $n, n+1, ..., N$ to objective function if the system starts in state s_n at stage n, we make decision x_n and we make optimal decisions at all future stages.

- Use the recursive relationship to work backward stage by stage.
Construct a table at each stage:

$s_n \backslash x_n$	$f_n(s_n, x_n)$	$f_n^*(s_n)$	x_n^*

Stages With Infinitely Many States

Continuous State Variables

Maximize

$$g(x, y) = 8x - 2x^2 + 144y - 3y^3$$

subject to

$$x + y \leq 7$$

$$x, y \geq 0$$

Solution via Dynamic Programming

Stages: Stage 1 = Choose x, Stage 2 = Choose y

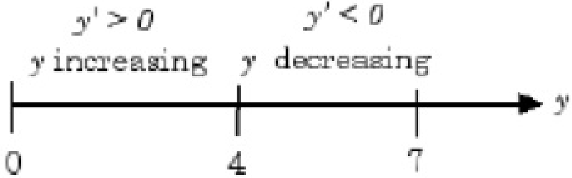
Let R = amount of slack left in constraint $x + y \leq 7$

$$g(x, y) = 8x - 2x^2 + 144y - 3y^3$$

Then $f_2^*(R) = \max\limits_{0 \leq y \leq R} (144y - 3y^3)$

$$f_2^*(R) = \max_{0 \le y \le R} (144y - 3y^3)$$

Now $f_2(R, y) = 144y - 3y^3$
so $\frac{\partial f_2}{\partial y} = 144 - 9y^2$
which equals 0 when $y = 4$.



Thus

$$y^* = \begin{cases} R & \text{if } 0 \le R \le 4 \\ 4 & \text{if } 4 \le R \le 7 \end{cases}$$

$$f_2^*(R) = \max_{0 \le y \le R} (144y - 3y^3)$$

$$y^* = \begin{cases} R & \text{if } 0 \le R \le 4 \\ 4 & \text{if } 4 \le R \le 7 \end{cases}$$

Note $f_2(R, 4) = 144(4) - 3(4^3) = 576 - 192 = 384$

n = 2

R	$f_2^*(R)$	y^*
$0 \le R \le 4$	$144R - 3R^3$	R
$4 \le R \le 7$	384	4

$$f_1^*(7) = \max_{0 \le x \le 7} \left(-2x^2 + 8x + f_2^*(7-x)\right)$$

$$= \max \begin{cases} \max\limits_{0 \le x \le 3}(-2x^2 + 8x + 384) \\ \max\limits_{3 \le x \le 7}(-2x^2 + 8x + 144(7-x) - 3(7-x)^3) \end{cases}$$

$\boxed{\text{For } 0 \leq x \leq 3}$

$$f_1(7,x) = -2x^2 + 8x + 384$$

Here $\frac{\partial f_1(7,x)}{\partial x} = -4x + 8$ which is 0 at $x = 2$

with maximum value $f_1(7,2) = -2(2^2) + 8(2) + 384 = 392$

$\boxed{\text{For } 3 \leq x \leq 7}$

$$f_1(7,x) = -2x^2 + 8x + 144(7-x) - 3(7-x)^3$$

Here $\frac{\partial f_1(7,x)}{\partial x} = -4x + 8 - 144 + 9(7-x)^2$

$$= -4x - 136 + 9(7-x)^2$$

The second derivative is $-4 - 18(7-x)$ which is negative so $f_1(7, x)$ will have a maximum value when its first derivative is 0.

$$9(7-x)^2 = 4x + 136$$

$$9(49 - 14x + x^2) - 4x - 136 = 0$$

$$9x^2 - 126x + 441 - 4x - 136 = 0$$

$$9x^2 - 130x + 305 = 0$$

Quadratic formula yields

$$x = \frac{130 \pm \sqrt{(-130)^2 - 36(305)}}{18}$$

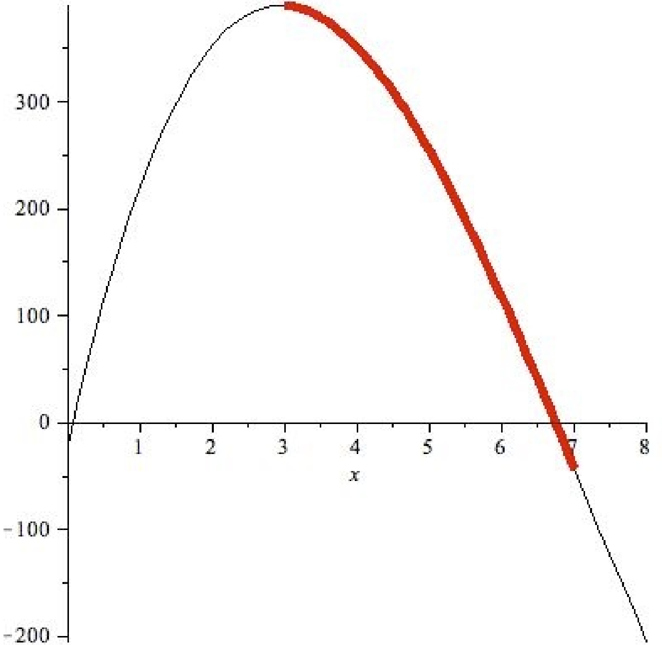
or $x = \frac{65 \pm 2\sqrt{370}}{9}$

Roots are approximately 11.5 and 2.95, neither of which is in the interval [3,7].
Thus the maximum occurs at an end point.

$$\text{Recall } \frac{\partial f_1(7,x)}{\partial x} = -4x + 8 - 144 + 9(7-x)^2$$

So at $x = 3 : -4(3) - 136 + 9 \times 4^2) = -12 - 136 + 144 < 0$
and at $x = 7 : -4(7_{-}136 + 9(0^2) = -28 - 136 < 0$
Thus $f_1(7,x)$ is decreasing at both endpoints.
The maximum occurs at $x = 3$.

Graph of $f_1(7, x)$



$$f_1(7,3) = -2(3^2) + 8(3) + 144(4) - 3(4^3) = 390$$

Thus $f_1^*(7) = \max(392, 390) = 392$ so $x_1^* = 2$.

Finally, with $x_1^* = 2, R = 7 - 2 = 5$ so $y^* = 4$.

The maximum value of the objective function is

$$g(x,y) = g(2,4) = 8(2) - 2(2^2) + 144(4) - 3(4^3) = 392.$$

Probabilistic Dynamic Programming I

Henry Brewster decides to market the Fancy Assortment through three local outlets: Shaw's, Hannaford, and the Middlebury Co-Op.

Each day he produces 6 cases (24 boxes each) at a cost to him of \$100 a case. Each case sold at an outlet yields \$200.

Any unsold assortments are returned to the factory where he can sell them at \$50 a case as stale products the next day.

Probability Distribution for Daily Demand
(demand in cases)

		1	2	3
Store 1	Shaw's	.6	0	.4
Store 2	Hannaford's	.5	.1	.4
Store 3	Middlebury Co-Op	.4	.3	.3

Problem: **How should he allocate the 6 cases to the three outlets to maximize his expected revenue?**

Observations

- Don't give more than 3 cases to any store
- Distribute all 6

Expected Revenue Earned from Allocating x_n cases to store n (in hundreds of dollars)

x_n	Store 1 Shaw's	Store 2 Hannaford's	Store 3 Co-Op
0	\$0	\$0	\$0
1	\$2	\$2	\$2
2	\$3.10	\$3.25	\$3.40
3	\$4.20	\$4.35	\$4.35

<u>Allocate 2 to Shaw's (Store 1)</u>

.6 (Sell 1, Return 1) + .4 (Sell 2)

$.6(\ 2 + 1/2\ ) + .4(4) = 1.5 + 1.6 = 3.1$

<u>Allocate 3 to Shaw's</u>

.6 (Sell 1, return 2) + .4(sell 3)

$.6(2 + 1) + .4(6) = 1.8 + 2.4 = 4.2$

Allocate 2 to Hannaford's (Store 2)

.5 (Sell 1, return 1) + .5(sell 2)

$.5(2 + 1/2) + .5(4) = 3.25$

Allocate 3 to Hannaford's

.5(sell 1, return 2) + .1(sell 2, return 1) + .4(sell3)

$.5(2 + 1) + .1(4 + 1/2) + .4(6)$

$1.5 + .45 + 2.4 = 4.35$

Allocate 2 to Co-Op

.4(sell 1, return 1) + .6(sell 2)

$.4(2 + 1/2) + .6(4) = 1 + 2.4 = 3.4$

Allocate 3 to Co-Op

.4(sell 1, return 2) + .3(sell 2, return 1) + .3(sell 3)

$.4(2 + 1) + .3(4 + ?) + .3(6) = 1.2 + 1.35 + 1.8 = 4.35$

Expected Revenue Earned from Allocating x_n cases to store n (in hundreds of dollars)

x_n	Store 1 Shaw's	Store 2 Hannaford's	Store 3 Co-Op
0	\$0	\$0	\$0
1	\$2	\$2	\$2
2	\$3.10	\$3.25	\$3.40
3	\$4.20	\$4.35	\$4.35

Let $r_i(s)$ = expected revenue of giving s cases to store i. Then the recursive relationship is

$$f_3(s) = r_3(s)$$
$$f_n(s) = r_n(s) + f^*_{n+1}(x - s)$$

$$f^*_n(s) = \max_{k \leq s}[r_n(k) + f^*_{n+1}(s - k)]$$

$$f_3(s) = r_3(s)$$

$n = 3$	Co-Op					
$s \backslash x_3$	**0**	**1**	**2**	**3**	f_3^*	x_3^*
0	**0**	–	–	–	**0**	**0**
1	**0**	**2**	–	–	**2**	**1**
2	**0**	**2**	**3.4**	–	**3.4**	**2**
3	**0**	**2**	**3.4**	**4.35**	**4.35**	**3**

$$f_2^*(s) = \max_{k \leq s}[r_2(k) + f_3^*(s-k)]$$

$n=2$	Hannaford					
s \ x_2	**0**	**1**	**2**	**3**	f_2^*	x_2^*
0	**0 +0 = 0**	–	–	–	**0**	**0**
1	**0 + 2 = 2**	**2 + 0 = 2**	–	–	**2**	**0 or 1**
2	**0+3.4=3.4**	**2 + 2 = 4**	**3.25 + 0**	–	**4**	**1**
3	**0+4.35=4.**	**2 +3.4= 5.4**	**3.25+2= 5.25**	**4.35 + 0 = 4.35**	**5.40**	**1**
4	–	**2 + 4.35=**	**3.25 + 3.4=6.65**	**4.35 + 2 = 6.35**	**6.65**	**2**
5	–	–	**3.25 + 4.35=7.60**	**4.35 + 3.4 =7.75**	**7.75**	**3**
6	–	–	–	**4.35+4.35 =8.70**	**8.70**	**3**

$$f_1^*(s) = \max_{k \le s}[r_1(k) + f_2^*(s-k)]$$

$n = 1$	Shaw's					
$s \backslash x_1$	**0**	**1**	**2**	**3**	f_1^*	x_1^*
6	**0 + 8.70 = 8.70**	**2 + 7.75 = 9. 75**	**3.10 +6.65 = 9.75**	**4.2 + 5.40 = 9.60**	**9.75**	**1 or 2**

Solutions

Store	Number of Cases	Number of Cases
Shaw's	1	2
Hannaford	3	2
Co-Op	2	2

Expected Value of Revenue: $975

Expected Value of Revenue: \$975

Best Possible Outcome

Sell all 6 cases $\rightarrow$ \$1200

Worst Possible Outcome

Sell 1 at each store

3 sales, 3 returns

Revenue $= 3 \cdot \$200 + 3 \cdot \$50 = \$750$

Next Time:

Use Dynamic Programming To:

Choose The Ideal Mate

Solve Fromage Cheese Company Problem


GOOD LUCK
FOR YOUR
EXAM
AND
DO THE BEST