Resource Allocation Problem Using Dynamic Programming

* You are given a fixed amount of resource and you want to allocate it among multible activities to maximize the total return.

Given:

(i) A total of R units of a resource
(ii) n activities ($A_1, A_2, \ldots A_n$).
(iii) Each activity A_i has return function $R_i(x)$, meaning it gives you return of $R_i(x)$ if you allocate x units to it.

Objective:

Find the best way to split the R units among the n activities so that total return is maximized.

How Dynamic Programming is used?

(i) Involves making repeated decisions (how much to give to each activity).
(ii) Has overlapping subproblems (split R across fewer activities).
(iii) Has optimal substructure (optimal solution uses optimal solution to subparts).

Intution:

(i) First solve for activity 1, then for 2 then 3 ...
(ii) At each stage, we compute the best return for all possible resource amounts upto R.
(iii) We reuse previously computed values instead of recalculating (memorization or tabulation).

Bellman's principle of optimality -

"An optimal policy has the property that, whatever the intial state and intial decision, the remaining decision must constitute an optimal policy with respect to the state resulting from first decision".

Bellman's Eqn for allocation:

$f_k(x)$ = max return from first k activities using x units

Then:

$$f_k(x) = \max_{0 \le x \le x} [R_k(x) + f_{k-1}(x-x)]$$

This recursive equation:

(i) Tries all possible values of x to allocate to the k^{th} activity

(ii) $x-x$ is what remains for previous $(k-1)$ activities

(iii) we find the max of all these combinations.

Example:

1.) Total resource = $x = 5$ units

2.) Number of activities = 2 (A_1, A_2)

Return tables:

units (x)	$R_1(x)$	$R_2(x)$
0	0	0
1	1	0
2	2	1
3	3	3
4	4	4
5	5	6

Step 1: Compute $f_1(x)$ → max return from A_1 using x units

since there is only one activity allocate all x units to A.

$$f_1(x) = R_1(x)$$

x	$f_1(x)$
0	0
1	1
2	2
3	3
4	4
5	5

Step 2: compute $f_2(x)$ - max return from A_1 and A_2 using x units:

we apply the bellman eqn

$$f_2(x) = \max_{0 \le n \le x} [R_2(n) + f_1(x-n)]$$

we now compute $f_2(x)$ for $x=0$ to 5 by trying all n values:

(i) $f_2(0)$

only possible $n = 0$

$R_2(0) + f_1(0) = 0 + 0$

$= 0$ ✓

(ii) $f_2(1)$

$n = 0 \rightarrow 0 + f_1(1)$

$= 0 + 1$

$= 1.$

$n = 1 \rightarrow 0 + f(0)$

$= 0.$

$f_2(1) = \max(1, 0)$

$= 1.$ ✓

(iii) $f_2(2)$

$x=0 \rightarrow 0 + f(2)$

$= 2$

$x = 1 \rightarrow 0 + f_1(1)$

$= 1$

$x = 2 \rightarrow 1 + f_1(0)$

$= 1$

$f_2(2) = \max(2, 1, 1) = 2$ ✓

(iv) $f_2(3)$:

$x=0 \rightarrow 0 + f_1(3) = 3$

$x=1 \rightarrow 0 + f_1(2) = 2$

$x=2 \rightarrow 1 + f_1(1) = 2$

$x=3 \rightarrow 3 + f_1(0) = 3$ ✓

$f_2(3) = \max(3, 2, 2, 3) = 3$ ✓

Similarly,

$f_2(4) = 4, \quad f_2(5) = 6$ ✓

Final DP table

x	$f_2(x)$	Best allocation
0	0	$n=0$
1	1	$n=0$
2	2	$n=0$
3	3	$n=0$ or 3
4	4	$n=0$ or 3 or 4
5	6	$n=5$

Computation method:

Bottom up tabulation.