

Here are two examples:

	Minute 1	Minute 2
A	2	10
B	1	20

The greedy algorithm would choose  $A$  for both steps, while the optimal solution would be to choose  $B$  for both steps.

	Minute 1	Minute 2	Minute 3	Minute 4
A	2	1	1	200
B	1	1	20	100

The greedy algorithm would choose  $A$ , then move, then choose  $B$  for the final two steps. The optimal solution would be to choose  $A$  for all four steps.

**(1b)** Let  $Opt(i, A)$  denote the maximum value of a plan in minutes 1 through  $i$  that ends on machine  $A$ , and define  $Opt(i, B)$  analogously for  $B$ .

Now, if you're on machine  $A$  in minute  $i$ , where were you in minute  $i - 1$ ? Either on machine  $A$ , or in the process of moving from machine  $B$ . In the first case, we have  $Opt(i, A) = a_i + Opt(i - 1, A)$ . In the second case, since you were last at  $B$  in minute  $i - 2$ , we have  $Opt(i, A) = a_i + Opt(i - 2, B)$ . Thus, overall, we have

$$Opt(i, A) = a_i + \max(Opt(i - 1, A), Opt(i - 2, B)).$$

A symmetric formula holds for  $Opt(i, B)$ .

The full algorithm initializes  $Opt(1, A) = a_1$  and  $Opt(1, B) = b_1$ . Then, for  $i = 2, 3, \dots, n$ , it computes  $Opt(i, A)$  and  $Opt(i, B)$  using the recurrence. This takes constant time for each of  $n - 1$  iterations, and so the total time is  $O(n)$ .

Here is an alternate solution. Let  $Opt(i)$  be the maximum value of a plan in minutes 1 through  $i$ . Also, initialize  $Opt(-1) = Opt(0) = 0$ . Now, in minute  $i$ , we ask: when was the most recent minute in which we moved? If this was minute  $k - 1$  (where perhaps  $k - 1 = 0$ ), then  $Opt(i)$  would be equal to the best we could do up through minute  $k - 2$ , followed by a move in minute  $k - 1$ , followed by the best we could do on a single machine from minutes  $k$  through  $i$ . Thus, we have

$$Opt(i) = \max_{1 \leq k \leq i} Opt(k - 2) + \max \left[ \sum_{\ell=k}^i a_{\ell}, \sum_{\ell=k}^i b_{\ell} \right].$$

The full algorithm then builds up these values for  $i = 2, 3, \dots, n$ . Each iteration takes  $O(n)$  time to compute the maximum, so the total running time is  $O(n^2)$ .

---

<sup>1</sup>ex803.497.915

A common type of error is to use a single-variable set of sub-problems as in the second correct solution (using  $Opt(i)$  to denote the maximum value of a plan in minutes 1 through  $i$ ), but with a recurrence that computed  $Opt(i)$  by looking only at  $Opt(i-1)$  and  $Opt(i-2)$ . For example, a common recurrence was to let  $m_j$  denote the machine on which the optimal plan for minutes 1 through  $j$  ended, let  $c(m_j)$  denote the number of steps available on machine  $m_j$  in minute  $j$ , and then write  $Opt(i) = \max(Opt(i-1) + c(m_{i-1}), Opt(i-2) + c(m_{i-2}))$ . But if we consider an example like

	Minute 1	Minute 2	Minute 3
A	2	10	200
B	1	20	100

then  $Opt(1) = 2$ ,  $Opt(2) = 21$ ,  $m_1 = A$ , and  $m_2 = B$ . But  $Opt(3) = 212$ , which does not follow from the recurrence above. There are a number of variations on the above recurrence, but they all break on this example.