Need help with a grid DP problem (select exactly K objects with shortest distance)

c-plus-plus dp dynamic-programming

bundishort Mar 24

Mar 24
1 / 11
Mar 24
Apr 6

Problem:

The kingdom is falling into ruin. People live in fear. Dragons pillage, kill, and just generally cause as much havoc as they possibly can. The king has just sent out a royal decree:

> *To any man out there who is able to bring me the heads of K dragons, I shall bequeath a lordship–to him, his sons and his grandsons, till the end of time.*

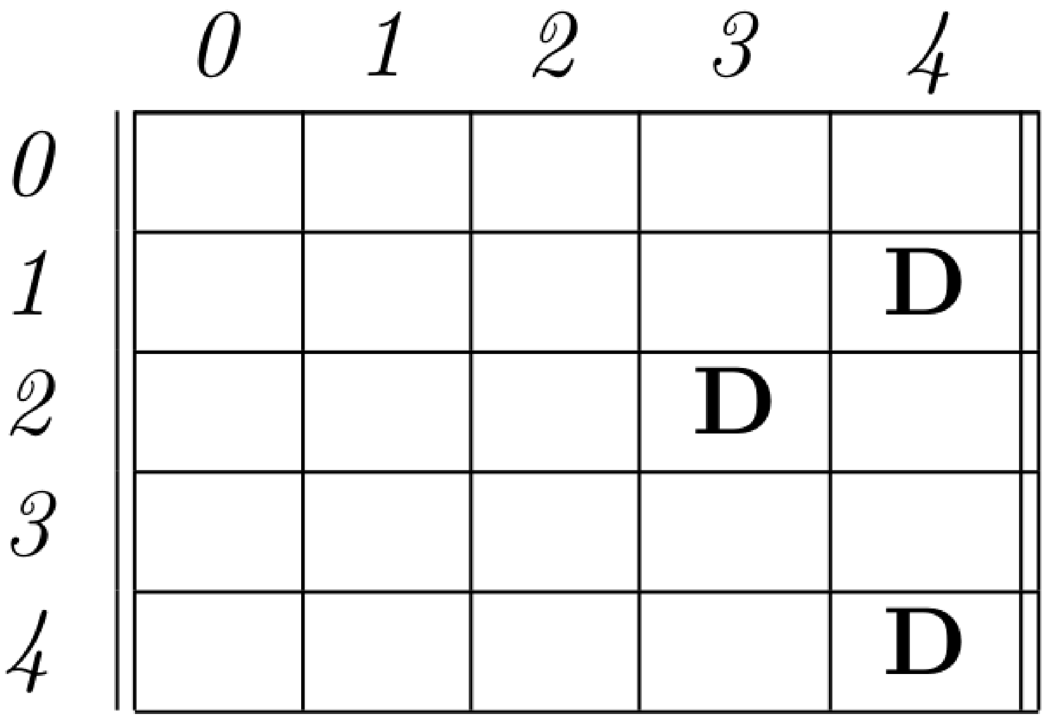
Having seen this royal decree, and knowing that you are capable of killing dragons thanks to your extensive medieval combat training, you set out on a quest to hunt down the evil creatures. Being a busy kind of guy, you would like to complete your quest quickly and kill K dragons through the shortest route.

The kingdom is arranged in a grid with R rows, numbered 0 to R-1, and C columns, numbered 0 to C-1 You start your quest at the top left corner of the grid, (0,0).

The total number of dragons in the kingdom is D, of which you have to kill K. Dragons are very territorial in nature, so each row of the grid contains at most one dragon. Also, since the kingdom is situated on a hill, you travel only downwards on the grid, though you may move left or right as you please.

You are told that no two dragons are on the same row of the grid. Also, no dragon is at position (0,0).

For example, suppose the grid has 5 rows and 5 columns with 3 dragons, of which you have to kill any 2. The three dragons are located at (1,4), (2,3) and (4,4), as shown below. In this case, your shortest route is to take 7 steps and kill the dragons in row 1 and row 2. Killing any other combination of 2 dragons takes 8 steps, so this is the minimum possible. Note that once you've killed K dragons, you don't incur any cost to return home. You just want to find how long it takes to do all the killing.



	0	1	2	3	4
0					
1					D
2				D	
3					
4					D

Sample input

```
5 5 2 3
1 4
4 4
2 3
```

Sample output

```
7
```

Sample input

```
5 5 3 4
1 1
3 2
2 4
0 4
```

Sample output

```
9
```

Sample input

```
10 10 5 7
1 7
3 6
5 1
6 5
7 4
4 3
2 2
```

Sample output

```
17
```

Input format

- Line 1 : Four space-separated integers, R, C, K and D.
- Lines 2 to D+1 : Each line has two-space separated integers r and c, the row and column of the corresponding dragon.

Output format

A single integer, the minimum total distance travelled to kill K dragons.

Test Data:

- In all testcases, K ≤ D ≤ R, and, for each dragon position (r,c), 0 ≤ r < R, and 0 ≤ c < C.
- In all testcases, 1 ≤ K,D ≤ 300.
- In 60% of the testcases, 1 ≤ R,C ≤ 300. In the remaining testcases, 1 ≤ R,C ≤ 100000.
- No two dragons will be on the same row.
- No dragon will be at position (0,0).

I thought of an approach but it is failing sample tests.
EDIT: Okay now my approach is passing 40% of the test cases. It's from a course so you'll won't be able to access the link
@ssjgz @everule1 @tmwilliamlin I am not expecting anything, but just in case y'all miss this.
I am sharing my approach below:
Consider the D'th dragon. It can either be included in our K-set or can't be. If it's not included then subproblem reduces to finding K dragons in D-1 dragons. If it's included then subproblem reduces to finding K-1 dragons in D-1 dragons. The distance increases by the Manhattan distance between co-ordinates of current dragon and the last dragon selected.

everule1 Mar 24

Question link and constraints?

bundishort Mar 24

I have now added the constraints

everule1 Mar 24

It's from 4 hours ago, so I'm gonna hope you're not cheating. Sort the dragons by rows.
Find the distance between all pairs of dragons and 0,0. Now define a dp state dp[i][j] being the minimum movement to kill j dragons and reach the ith dragon. Dp[i][j] is the minimum of dp[s][j-1] + dist[s][i] for all s less than i. This solves it in O(d^2 * k).

bundishort Mar 24

> bundishort:
>
> Consider the D'th dragon. It can either be included in our K-set or can't be. If it's not included then subproblem reduces to finding K dragons in D-1 dragons. If it's included then subproblem reduces to finding K-1 dragons in D-1 dragons. The distance increases by the Manhattan distance between co-ordinates of current dragon and the last dragon selected.

Thanks. I'll try to understand your solution. Any idea why this might be incorrect though?

everule1 Mar 24