CSE 431/531 Analysis of Algorithms
Problem Set 4

Chen Xu

Due Date: August 3, 2023 23:59 PM EST.

Problem 1. (40%)

Now here is another chance of you share your problems. We have learned about Dynamic Programming. There are thousands of problems and variants out there. Here we ask you to complete the following tasks:

1. Accurately state your problem. The problem should at least involve Dynamic Programming, additionally, you can add Greedy or Divide and Conquer on top of that.
2. Write the solution for your own problem.

Here is what we are looking for:

- (coefficient $a \leq 4\%$) The problem should be accurate. This means that everything is defined mathematically. The problem does not need to be long.
- (coefficient $c_1 \leq 2.0$) We want to be as creative as possible. For this problem you are allowed to look up the internet. Of course, the instructor will also look up the internet to judge if your problem is novel. Bonus points (2%) if your problem is based on real life situation. Unanswered stackexchange and mathoverflow posts etc. still count as novel. We suggest starting from a well known problem and making alternations. But if the alternation makes no significant difference to the original solution then you will lose some points.

- (coefficient $c_2 \leq 2.0$) The problem should be challenging. We will test if AI can answer your problems. You may test yourself as well. The instructor will pick the top 5 most interesting ones to give bonus (5%).
- (coefficient $b \leq 18\%$) The solution must be complete. This includes correctness, time and space complexity.

The score is evaluated by this formula: $(c_1 + c_2) * b + a + bonus$

You have the copyrights of your own problem. It is possible that some variants of your problem may appear as a question in the final exam random question pool. In your response, please indicate if you would like to share with the class and authorize the instructor to change and put your problem in the final exam. If so, then you need to write your response in .tex/.doc.

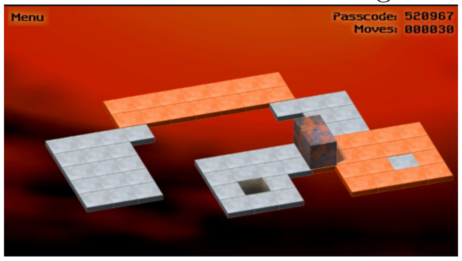
Problem 2. (30%)

In the last Homework problem 3 we had the max score from top left to bottom right problem. Now we are restricting the number of 90-*degree turns* you can make upto k times (k is a part of input too.). A 90-*degree turn* is defined as the pair of consecutive moves that is in different directions, for example, last move you moved down but this move you moved right so you made a 90-degree turn. Design an algorithm to compute the path you take that maximizes this score under the restriction of k turns. State your recursive structure.

Problem 3. (30%)

A Bloxorz game is an interesting block moving game. You are given a $n \times n$ grid and want to move a $1 \times 1 \times 2$ block in a specific way to reach and fall through the destination 1×1 hole. Here is a link to the game: `https://www.coolmathgames.com/0-bloxorz`.

Here is a screenshot of the game:



We want to simplify the our version of Bloxorz.

The position of the block can be *standing* or *flat*. In standing position, the block takes one cell space. In flat position, the block takes two cell space.

The cells of the grid can be

- 0 – empty cell. A block can fall on that cell.
- 1 – floor cell. A block is supported on that cell.
- 2 – destination cell. This is our target. A block can fall on that cell. But to win the game, the block must fall when it is in standing position. There will be only one destination cell.
- 3 – starting cell. A block is placed in standing position on that cell initially. There will be only one starting cell.

In standing position, you can roll the block to the adjacent cells. The block will then lay flat. In flat position, when you roll it by the long edge, it will roll up to the adjacent cell and stay flat. When you roll it by the short edge, it will roll up then become standing again.

The block falls when:

- In standing position and the cell is empty or destination.
- In flat position, the two cells are both empty.
- In flat position, one of the cells is empty and the outer cell adjacent to the empty cell is also empty. i.e. $\boxed{10}0$

The block will not fall in flat position when one of its cell is empty but the outer cell is floor (It gets supported. i.e. $\boxed{10}1$).

Let us be more restrictive to say that the grid is surrounded by double layer 0s so that it is guaranteed to fall on the edges of the grid.

Given a grid, design an efficient algorithm to compute the minimum number of rolls to beat the level. Explain how and why it works and how efficient the algorithm is.

(Hint1: Some level is unbeatable)

(Hint2: It is equivalent to some graph problem)