

Type to search

LeetCode

Introduction

Algorithms

Depth-first Search

Breadth-first Search

Union Find

Tree

Dynamic Programming

Topological Sort

Substring Problem Template

Kadane's Algorithm

KMP

Fenwick Tree or Binary Indexed Tree

Segment Tree Range Minimum Query


Subset Sum

Sweep-line Algorithm

Expression Tree

Interval Search Tree

Monte Carlo Method

 Powered by **GitBook**

☰

A

🐦

f

🔗

188. Best Time to Buy and Sell Stock IV (Hard)

Say you have an array for which the i^{th} element is the price of a given stock on day i .

Design an algorithm to find the maximum profit. You may complete at most k transactions.

Note:

You may not engage in multiple transactions at the same time (ie, you must sell the stock before you buy again).

Solution 1: 6ms

这道题实际上是之前那道 **Best Time to Buy and Sell Stock III** 买股票的最佳时间之三的一般情况的推广，还是需要用动态规划Dynamic programming来解决，具体思路如下：

这里我们需要两个递推公式来分别更新两个变量local和global，参见网友Code Ganker的博客，我们其实可以求至少k次交易的最大利润。我们定义local[i][j]为在到达第i天时最多可进行j次交易并且最后一次交易在最后一天卖出的最大利润，此为局部最优。然后我们定义global[i][j]为在到达第i天时最多可进行j次交易的最大利润，此为全局最优。它们的递推式为：

$$local[i][j] = \max(global[i - 1][j - 1] + \max(diff, 0), local[i - 1][j] + diff)$$
$$global[i][j] = \max(local[i][j], global[i - 1][j]),$$

其中局部最优值是比较前一天并少交易一次的全局最优加上大于0的差值，和前一天的局部最优加上差值后相比，两者之中取较大值，而全局最优比较局部最优和前一天的全局最优。

但这道题还有个坑，就是如果k的值远大于prices的天数，比如k是好几百万，而prices的天数就为若干天的话，上面的DP解法就非常的没有效率，应该直接用Best Time to Buy and Sell Stock II 买股票的最佳时间之二的的方法来求解，所以实际上这道题是之前的二和三的综合体，代码如下：

```
class Solution {
```