ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт по лабораторной работе № 6

«Динамическое программирование»

Выполнил работу:

Мавров Артём

Академическая группа №J3112

Принято:

Ассистент (квалификационная категория "ассистент"), Дунаев Максим

Санкт-Петербург

2024

Введение

Цель лабораторной работы — на практике изучить принципы динамического программирования.

Задачи – изучить динамическое программирование, применить знания на практике решив hard задачу на leetcode.

**Теоретическая подготовка:**

Для реализации этой лабораторной работы потребовалось:

* Базовые знания языка C++
* Знания динамического программирования

**Реализация:**

* Реализация задачи с leetcode: https://leetcode.com/problems/best-time-to-buy-and-sell-stock-iv/

Экспериментальная часть

**Реализация кода, подсчёт асимптотики:**

#include <iostream>

#include <vector>

#include <algorithm>

#include <climits>

using namespace std;

class Solution {

public:

    int maxProfit(int k, vector<int>& prices) {

        int n = prices.size();

        if (n == 0 || k == 0) {

            return 0;

        }

        if (k >= n / 2) {

            int max\_profit = 0;

            for (int i = 1; i < n; i++) { //O(n)

                if (prices[i] > prices[i - 1]) {

                    max\_profit += prices[i] - prices[i - 1];

                }

            }

            return max\_profit;

        }

        vector<vector<int>> dp(k + 1, vector<int>(n, 0));

        for (int j = 1; j <= k; j++) { //O(k)

            int max\_profit\_prev = INT\_MIN;

            for (int i = 1; i < n; i++) { //O(n)

                max\_profit\_prev = max(max\_profit\_prev, dp[j - 1][i - 1] - prices[i - 1]);

                dp[j][i] = max(dp[j][i - 1], max\_profit\_prev + prices[i]);

            }

        }

        return dp[k][n - 1];

    }

};

//Итого O(n) в лучшем случае(при k >= n / 2), O(k \* n) в худшем (при k < n)

(приложение 1: решение задачи)

**Подсчёт пространственной сложности:**

Используется вектор векторов O(n \* k) и незначительные переменные. Итого пространственная сложность O(n \* k).

**Обоснование использования ДП:**

Данная задача может быть решена с помощью рекурсии, но тогда асимптотика будет O(2^n) т.к. на каждом шаге будет два варианта. Возможна оптимизация при рассматривании только пиков и впадин, но это уменьшит сложность незначительно. Использование ДП позволяет значительно ускорить работу алгоритма за счёт использования таблицы ДП.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание(приложение 2: решение задачи на leetcode)