Московский Авиационный Институт
(Национальный Исследовательский Университет)
Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Лабораторная работа №7 по курсу

«Дискретный анализ»

Студент: Ибрагимов Р. Р.
Группа: М8О-303Б-22
Вариант: 2
Преподаватель: Макаров Н.К.
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2024

Постановка задачи
У вас есть рюкзак, вместимостью m, а так же n предметов, у каждого из которых есть вес w_i и стоимость c_i. Необходимо выбрать такое подмножество I из них, чтобы:
1. $(\Sigma w\_i) \leq m$, где $i \in I$
2. $(\Sigma c\_i) * |I|$, где $i \in I$ является максимальной из всех возможных.
$|I|$ – мощность множества I.

Формат ввода
В первой строке заданы $1 \leq n \leq 100$ и $1 \leq n \leq 5000$. В последующих n строках через пробел заданы параметры предеметов: w_i и c_i.

Формат вывода
В первой строке необходимо вывести одно число – максимальное значение $(\Sigma c\_i) * |I|$, где $i \in I$, а на второй – индексы предметов, входящих в ответ.

Метод решения
Задачу будем решать с помощью динамического программирования. dp[i][j][k] — максимальная стоимость j предметов из первых i предметов в рюкзаке вместимости k. Под стоимостью, согласно условию задачи, понимается суммарная стоимость предметов, умноженная на их количество. Переход динамики можно описать следующим образом: dp[i][j][k] = max(dp[i - 1][j][k], dp[i - 1][j - 1][k - w[i - 1]]/(j - 1) + c[i - 1]) * j.
При реализации сначала будут вычисляться dp[i][j][k] при j = 1 (берём один предмет), затем уже будет учитываться, что можно брать два и более предметов. Ответом будет максимальный элемент по всем j при i = n и k = m (в dp[n][j_max][m], где j_max индекс, по которому располагается максимум). В конце потребуется восстановить ответ.
Сложность алгоритма решения: O(n^2 * m), где n — количество предметов, m — вместимость рюкзака.

Исходный код
```
#include <iostream>
#include <vector>
#include <algorithm>

using ul = unsigned long;

int main() {
    std::ios::sync_with_stdio(false);
    std::cin.tie(0);
    std::cout.tie(0);

    size_t n, m;

    std::cin >> n >> m;
```

```
    std::vector<ul> w(n);
    std::vector<ul> c(n);

    for (size_t i = 0; i < n; ++i) {
        std::cin >> w[i] >> c[i];
    }

    std::vector<std::vector<std::vector<ul>>> dp(n + 1, std::vector<std::vector<ul>>(n + 1,
std::vector<ul>(m + 1, 0)));

    for (size_t i = 1; i < n + 1; ++i) {
        for (size_t j = 1; j < n + 1; ++j) {
            for (size_t k = 1; k < m + 1; ++k) {
                dp[i][j][k] = dp[i - 1][j][k];

                if (k >= w[i - 1]) {
                    if (j == 1) {
                        dp[i][j][k] = std::max(dp[i][j][k], dp[i - 1][j][k - w[i - 1]] + c[i - 1]);
                    }
                    else if (dp[i - 1][j - 1][k - w[i - 1]] > 0) {
                        dp[i][j][k] = std::max(dp[i][j][k], (dp[i - 1][j - 1][k - w[i - 1]] / (j - 1) + c[i - 1]) * j);
                    }
                }
            }
        }
    }

    ul max_value = 0;
    size_t j_max = 0;
    for (size_t j = 0; j < n + 1; ++j) {
        if (dp[n][j][m] > max_value) {
            max_value = dp[n][j][m];
            j_max = j;
        }
    }

    std::vector<size_t> path;
    while (dp[n][j_max][m] > 0) {
        if (dp[n][j_max][m] == dp[n - 1][j_max][m]) {
            --n;
        }
        else {
            m -= w[n - 1];
            --j_max;
            path.push_back(n--);
        }
    }

    std::reverse(path.begin(), path.end());

    std::cout << max_value << '\n';
    for (size_t elem : path) {
```

```
        std::cout << elem << ' ';
    }
    std::cout << '\n';

    return 0;
}
```

Тесты

Номер теста	Ввод	Вывод
1	3 6 2 1 5 4 4 2	6 1 3
2	5 130 76 2 62 10 19 67 36 78 26 90	705 3 4 5
3	7 31 78 94 28 19 46 46 16 32 35 48 12 65 78 61	194 4 6

Выводы

В ходе выполнения данной лабораторной работы был более подробно изучен подход динамического программирования и, в частности, вариация задачи о рюкзаке. Сама идея разбивания большой подзадачи на мелкие задачи весьма интересна и интуитивно понятна, но для себя отмечу, что для меня даётся труднее всего.