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: А. Ю. Голов
Преподаватель: С. А. Михайлова
Группа: М8О-301Б-21

Дата:
Оценка:
Подпись:

Москва, 2023

Лабораторная работа №1

Формулировка задания: Задана матрица натуральных чисел A размерности $m \times n$. Из текущей клетки можно перейти в любую из 3-х соседних, стоящих в строке с номером на единицу больше, при этом за каждый проход через клетку $(i,\ j)$ взымается штраф A_{ij}. Необходимо пройти из какой-нибудь клетки верхней строки до любой клетки нижней, набрав при проходе по клеткам минимальный штраф.

Формат ввода: Первая строка входного файла содержит в себе пару чисел $2 \leq n,\ m \leq 1000$ и , затем следует n строк из m целых чисел.

Формат вывода: Необходимо вывести в выходной файл на первой строке минимальный штраф, а на второй – последовательность координат из n ячеек, через которые пролегает маршрут с минимальным штрафом.

1 Описание

"В общем случае мы можем решить задачу, в которой присутствует оптимальная подструктура, проделывая следующие три шага:

1. Разбиение задачи на подхадачи меньшего размера.
2. Нахождение оптимального решения подзадач рекурсивно, проделывая такой же трёхшаговый алгоритм.
3. Использование полученного решения подзадач для конструирования решения исходной задачи.

Подзадачи решаются делением их на подзадачи ещё меньшего размера и т.д., пока не приходят к тривиальному случаю задачи, решаемой за константное время (ответ можно сказать сразу).

В данной задаче мы будем делать проход снизу вверх, последовательно считая минимальный штраф для попадания в текущую клетку массива для всех клеток на основе уже посчитанных таким образом минимальных штрафов для клеток внизу. Очевидно, что для клеток со строки n-1 ничего считать не нужно.

После этого их первой строки выбирается клетка с наименьшим штрафом, а затем алгорит проходит вниз по теблице, выбирая наименьшее число из трёх клеток снизу.

2 Исходный код

```
#include <bits/stdc++.h>
#include <istream>
#include <vector>

std::istream& operator >> (std::istream& in, std::vector<std::vector<int64_t>> &data)
    {
  for (int64_t i = 0; i < (int64_t)data.size(); ++i){
    for (int64_t j = 0; j < (int64_t)data[0].size(); ++j){
      in >> data[i][j];
    }
  }

  return in;
}

void DisplayAnswer(std::vector<std::vector<int64_t>> &data)
{
  int64_t n = data.size();
  int64_t m = data[0].size();

  int64_t minValue = data[0][0];
  int64_t idx = 0;

  for (int64_t i = 1; i < m; ++i) {
    if (data[0][i] <= minValue) {
      minValue = data[0][i];
      idx = i;
    }
  }

  std::cout << minValue << '\n';

  int64_t i, j = idx;

  for (i = 0; i < n - 1; ++i)
  {
    std::cout << '(' << i + 1 << "," << j + 1 << ") ";

    int64_t tmp;

    if (j == 0) {
      tmp = std::min(data[i + 1][j], data[i + 1][j + 1]);
    }
    else if (j == m - 1) {
      tmp = std::min(data[i + 1][j - 1], data[i + 1][j]);
    }
```

```
46      else {
47        tmp = std::min({data[i + 1][j - 1], data[i + 1][j], data[i + 1][j + 1]});
48      }
49
50      if (tmp == data[i + 1][j + 1]) {
51        j++;
52      }
53      else if (tmp == data[i + 1][j - 1]) {
54        j--;
55      }
56    }
57
58    std::cout << '(' << i + 1 << "," << j + 1 << ")\n";
59  }
60
61  int main()
62  {
63    int64_t n, m;
64    std::cin >> n >> m;
65
66    std::vector<std::vector<int64_t>> data (n, std::vector<int64_t>(m));
67
68    std::cin >> data;
69
70    /*
71    for (int64_t i = 0; i < n; ++i) {
72      for (int64_t j = 0; j < m; ++j) {
73        std::cin >> data[i][j];
74      }
75    }
76    */
77
78    for (int64_t i = n - 2; i > -1; --i) {
79      for (int64_t j = 0; j < m; ++j)
80      {
81        if (j == 0) {
82          data[i][j] += std::min(data[i + 1][j], data[i + 1][j + 1]);
83        }
84        else if (j == m - 1) {
85          data[i][j] += std::min(data[i + 1][j - 1], data[i + 1][j]);
86        }
87        else {
88          data[i][j] += std::min({data[i + 1][j - 1], data[i + 1][j], data[i + 1][j +
                1]});
89        }
90      }
91    }
92
93    DisplayAnswer(data);
```

```
94 ||
95 ||   return 0;
96 || }
```

3 Тест производительности

В качестве тестирующего инструмента были выбраны гугл-тесты, интегрированные в программу при помощи CMake. Применён метод «table-driven tests».

4 Выводы

По итогам выполнения седьмой лабораторной работы по курсу «Дискретный анализ», стало очевидно, что грамотный практический подход к задаче ведёт к оптимизации производительности программы посредством применения верного принципа разработки программы.

Список литературы

[1] Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. *Алгоритмы: построение и анализ, 2-е издание.* — Издательский дом «Вильямс», 2007. Перевод с английского: И. В. Красиков, Н. А. Орехова, В. Н. Романов. — 1296 с. (ISBN 5-8459-0857-4 (рус.))

[2] *Сортировка подсчётом — Википедия.*
URL: `http://ru.wikipedia.org/wiki/Сортировка_подсчётом` (дата обращения: 16.12.2013).

[3] Список использованных источников оформлять нужно по ГОСТ Р 7.05-2008