Московский авиационный институт
(национальный исследовательский университет)

Факультет компьютерных наук и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №8 по курсу «Дискретный анализ»

Студент: С. Ю. Свиридов
Группа: М8О-306Б-22
Дата:
Оценка:
Подпись:

Москва 2025

Жадные алгоритмы

Задача: С. 3 Максимальный треугольник

Заданы длины N отрезков, необходимо выбрать три таких отрезка, которые образовывали бы треугольник с максимальной площадью.

Формат ввода

На первой строке находится число N, за которым следует N строк с целыми числами-длинами отрезков.

Формат вывода

Если никакого треугольника из заданных отрезков составить нельзя – 0, в противном случае на первой строке площадь треугольника с тремя знаками после запятой, на второй строке – длины трёх отрезков, составляющих этот треугольник. Длины должны быть отсортированы.

1 Описание

Для решения задачи используем жадный алгоритм. Жадные алгоритмы (или жадные методы) - это подход к решению задач, при котором на каждом шаге принимается решение, основанное на локально оптимальном выборе, с надеждой, что это приведет к глобально оптимальному решению. В данной задаче предлагается просто вычислять площади треугольников с помощью формулы Герона (используя полупериметр), проверяя перед этим возможность их существования. Площадь будем вычислять, начиная с самых длинных отрезков, однако будем проверять все возможные случаи.

Реализация

Сортируем все длины отрезков в порядке убывания. Далее циклом, начиная с начала перебираем все возможные тройки длин отрезков и вычисляем площади треугольников, которые можно составить. Обновляем максимальную площадь и запоминаем длины отрезков, из которых составили треугольник; после полного обхода возвращаем максимальную площадь и длины отрезков.

Проверять возможность существования треугольника будем по условию $a < b + c$. (Длина стороны должны быть меньше суммы двух других сторон). Формула для вычисления площади - $sqrt(p * (p - a) * (p - b) * (p - c))$, где $p = (a + b + c)/2$ - полупериметр треугольника

2 Исходный код

Приложен исходный код программы.

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <cmath>
#include <iomanip>

double calculate_area(int a, int b, int c) {
    double s = (a + b + c) / 2.0;
    return std::sqrt(s * (s - a) * (s - b) * (s - c));
}

std::vector<int> read_segments(int N) {
    std::vector<int> segments(N);
    for (int& segment : segments) {
        std::cin >> segment;
```

```
16      }
17      return segments;
18  }
19
20  bool is_valid_triangle(int a, int b, int c) {
21      return a < b + c;
22  }
23
24  std::tuple<double, int, int, int> find_max_area_triangle(const std::vector<int>&
        segments) {
25      double max_area = 0.0;
26      int a = 0, b = 0, c = 0;
27
28      for (size_t i = 0; i < segments.size() - 2; ++i) {
29          if (is_valid_triangle(segments[i], segments[i + 1], segments[i + 2])) {
30              double area = calculate_area(segments[i], segments[i + 1], segments[i + 2])
                    ;
31              if (area > max_area) {
32                  max_area = area;
33                  a = segments[i + 2];
34                  b = segments[i + 1];
35                  c = segments[i];
36              }
37          }
38      }
39
40      return {max_area, a, b, c};
41  }
42
43  void print_result(double max_area, int a, int b, int c) {
44      if (max_area == 0.0) {
45          std::cout << "0" << std::endl;
46      } else {
47          std::cout << std::fixed << std::setprecision(3) << max_area << std::endl;
48          std::cout << a << " " << b << " " << c << std::endl;
49      }
50  }
51
52  int main() {
53      int N;
54      std::cin >> N;
55
56      auto segments = read_segments(N);
57      std::sort(segments.rbegin(), segments.rend());
58
59      auto [max_area, a, b, c] = find_max_area_triangle(segments);
60      print_result(max_area, a, b, c);
61
62      return 0;
```

```
63 || }
```

3 Консоль

```
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ g++ main.cpp
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
4
1
2
3
5
0
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
3
14
24
34
137.870
14 24 34
```

Сложность алгоритма - $O(NlogN)$

4 Тест производительности

В тестах будет произведен расчет площади максимального треугольника, составленного из длин треугольников, которые хранятся в массивах длиной 10, 100, 1000, 10000, 100000 и 1000000 элементов.

```
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ g++ bench.cpp
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
Select count of lines: 10
Result time: 4 ms
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
Select count of lines: 100
Result time: 23 ms
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
Select count of lines: 1000
Result time: 321 ms
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
Select count of lines: 10000
Result time: 3459 ms
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
Select count of lines: 100000
Result time: 47248 ms
potatogrill24@DESKTOP-7CM71EV:~/progs/Diskran/laba8$ ./a.out
Select count of lines: 1000000
Result time: 509919 ms
```

Точность действительно близка к линеарифметической, особенно это заметно на больших количествах длин отрезков в массиве.

5 Выводы

В ходе выполнения данной лабораторной работы я познакомился с жадными алгоритмами. Я воспользовался формулой Герона, написал простой код и проанализировал работу моего жадного алгоритма.

Список литературы

[1] *Формула Герона*
URL: https://ru.wikipedia.org/wiki/Формула_Герона

[2] *Жадные алгоритмы*
URL: https://habr.com/ru/articles/120343/