ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт по лабораторной работе № 4

«Кластеризация массива посредством полного перебора всех комбинаций значений этого массива»

Выполнил работу

Лапшин Максим

Академическая группа J3111

Принято

Практик, магистр Вершинин Владислав

Санкт-Петербург

2024

**Структура отчёта:**

**Введение:**

Цель работы: реализовать алгоритм кластеризации массива методом полного перебора всех возможных разбиений с минимизацией отклонения элементов от среднего значения подмассивов.

**Задачи:**

Разработать алгоритм полного перебора для поиска оптимального разбиения.

Реализовать функцию вычисления стоимости разбиения.

Протестировать алгоритм на различных входных данных.

Теоретическая подготовка

**Для выполнения задачи необходимо учитывать:**

**Тип данных:** массивы чисел, включающие целые и вещественные значения, как положительные, так и отрицательные.

**Метрика оптимизации:** минимизация суммы абсолютных отклонений элементов подмассива от их среднего значения.

**Алгоритм:** полный перебор всех возможных разбиений с проверкой корректности и вычислением стоимости.

Сложность алгоритма полного перебора составляет

O(N2^N), где N — количество элементов в массиве.

Каждый бит в маске определяет, является ли соответствующий элемент массива концом текущего кластера. Например, для массива из 4 элементов маска 101 задаёт разбиение как(1),(2,3),(4) т.к. последний этемент всегда 1 то маска N - 1. сложность O(2^(N-1))

Алгоритм полного перебора имеет сложность 𝑂(𝑁)

Дополнительно используются операции вычисления средней величины и оценки стоимости, сложность которых составляет O(N).

Отсюда и сложность O(N2^N)

**Реализация**

**Этапы выполнения работы:**

**Сортировка массива:** для оптимизации процесса обработки и упрощения вычислений. Реализована простая сортировка с квадратичной сложностью 𝑂(𝑁\*2).

**Вычисление стоимости разбиения:** функция allcost оценивает суммарное отклонение элементов от среднего значения для заданного разбиения.

**Перебор всех комбинаций разбиений:** с использованием битового представления возможных вариантов.

**Фрагменты кода:**

**Функция сортировки массива:**

void sort(vector<double>& a) {

for (int i = 0; i < a.size(); i++) {

double min = a[i];

int min\_ind = i;

for (int j = i + 1; j < a.size(); j++) {

if (a[j] < min) {

min = a[j];

min\_ind = j;

}

}

swap(a[i], a[min\_ind]);

}

}

**Вычисление стоимости разбиения:**

double allcost(const vector<double>& arr, const vector<int>& nu, int n) {

double ans = 0;

double cur = 0;

int f = 0;

for (int i = 0; i < n-1; ++i) {

if (nu[i] == 0) {

cur += arr[i];

f += 1;

} else {

cur += arr[i];

double h = cur / (f + 1);

for (int g = 0; g <= f; ++g) {

ans += abs(arr[i - f + g] - h);

}

cur = 0;

f = 0;

}

}

return ans;

}

**Экспериментальная часть**

**Тестирование проводилось на массивах различной длины.**

**Таблица 1 — Результаты тестирования алгоритма**



**Рисунок 1 —** График зависимости времени выполнения от числа элементов

**Анализ:**

Экспериментальные данные соответствуют теоретическим оценкам сложности.

Алгоритм эффективно обрабатывает массивы небольшого размера.

**Заключение**

В ходе выполнения работы был реализован алгоритм кластеризации массива методом полного перебора. Цель работы достигнута, алгоритм протестирован на массивах различных размеров.

**Перспективы дальнейшей работы:**

Оптимизация алгоритма за счёт сокращения перебираемых комбинаций.

Реализация параллельных вычислений для обработки больших массивов**.**