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

Косарев И. А.

Академическая группа №J3113

Принято

Ходненко И. В.

Санкт-Петербург

2024

**Структура отчёта:**

1. Введение

Цель: понять, насколько затратными могут быть решения задач алгоритмами высокой сложности, с помощью реализации алгоритма кластеризации путем полного перебора всех возможных кластеров (каждый кластер на выходе должен содержать как минимум один элемент) по метрике минимизации расстояния между каждым элементом кластера и средним значением в кластере с сохранением порядка (кол-во кластеров – 6).

Задачи:

* + - 1. Написать итерируемую структуру, которая создает маски кластеризации.
      2. Написать вспомогательную функцию, которая высчитывает метрику для конкретного разбиения.
      3. Написать саму функцию кластеризации, которая с помощью полного перебора масок возвращает наилучшее разбиение массива на 6 кластеров.
      4. Написать тесты для проверки работоспособности алгоритма.
      5. Оценить полученный алгоритм по памяти и асимптотике.

1. Теоретическая подготовка

Для выполнения лабораторной работы необходимо знание понятий кластеризации, маски кластеризации, O-нотации, работы со ссылками и структурами на языке C++. Необходимо также знание библиотек chrono (для подсчета времени), vector (для задания динамических массивов), limits (для оценки метрики разбиения на кластеры) и типов данных int, double, vector<>, bool.

1. Реализация

Для начала необходимо реализовать итерируемую структуру, описанную в задачах выше. По факту, маска кластеризации, задающая конкретное разбиение массива на кластеры, является числом в шестеричной системе счисления. Следовательно, чтобы рассмотреть всевозможные разбиения, необходимо рассмотреть все маски, следовательно, все числа в шестеричной системе в определенном диапазоне. Итерацией (переход от маски к следующей, метод get\_next()) является взятие следующего числа в шестеричной системе (хранение числа реализовано массивом длиной n, где n – кол-во элементов). Необходимо также реализовать проверку маски на соответствие условию «в каждом кластере должен быть по крайней мере один элемент» (метод check\_mask()). Принцип работы представлен на рисунке 1.

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

Автоматически созданное описание

Рисунок 1 - структура cluster\_masks

Далее необходимо реализовать функцию подсчета метрики для конкретной маски. Принцип работы представлен ниже на рисунке 2.

Изображение выглядит как снимок экрана, текст

Автоматически созданное описание

Рисунок 2 - функция get\_score()

Наконец, реализуем саму функцию кластеризации. С помощью итерируемой структуры cluster\_masks рассмотрим каждое возможное разбиение на кластеры и подсчитаем метрику, одновременно обновляя наилучшие разбиение и метрику. В результате функция возвращает массив кластеров (разбиение наилучшим образом). Ниже на рисунке 3 представлен принцип работы.

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 3 - функция clusterize()

Для проверки принципа работы напишем тесты с помощью тестирующего фреймворка doctest: рассмотрим случаи, когда разбиение невозможно априори, когда требуется разбиение массива с одинаковыми элементами и общий случай.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 4 – тесты

1. Экспериментальная часть

Подсчет асимптотики и памяти каждой структуры и функции представлены на рисунках 1-4 (значения расхода памяти указаны в байтах), в общем случае затраты – 147 + 14\*n байт, где n – кол-во элементов в изначальном массиве.

Большинство функций и методов имеют линейную асимптотику (O(n)), общая асимптотика (рассмотрения каждой маски, ее проверки и вычисления метрики для нее) – O(n \* 6 ^ n) (расписано подробнее на рисунке 3).

Для тестирования алгоритма была собрана статистика, приведенная в таблице №1 (кол-во элементов входного массива начинается с 6, так как есть условие, что каждый кластер должен содержать хотя бы 1 элемент; также возьмем в рассмотрение условную асимптотику O(6^n \* n \* 4) (теоретически большую O(n^6 \* n) в 4 раза) для того, чтобы показать, что полученная асимптотика алгоритма лежит в пределах O(6^n \* n) с точностью до умножения на константу).

Таблица №1 - Подсчёт сложности реализованного алгоритма

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Размер входного набора | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| Время выполнения программы, с | 0.002 | 0.028 | 0.184 | 0.505 | 3.744 | 24.811 | 181.895 |
| O(6^n \* n), с | ~0 | 0.002 | 0.023 | 0.127 | 0.869 | 7.463 | 49.628 |
| O(6^n \* n \* 4), с | 0.002 | 0.013 | 0.107 | 0.646 | 4.146 | 26.97 | 179.523 |

График представляющий визуально удобный формат данных из таблицы №1 представлен на рисунке 5.

Рисунок 5 - сравнение времени выполнения

Из полученных данных видно, что асимптотика алгоритма действительно O(6 ^ n \* n), что соответствует предположению (графики времени выполнения программы и O(n ^ 6 \* n \* 4) почти полностью совпадают).

1. Заключение

В ходе выполнения работы мною был реализован алгоритм кластеризации массива вещественных чисел по метрике минимального расстояния до среднего значения в кластере. Цель работы была достигнута путём тестирования на различных наборах входных данных. Полученные результаты также совпадают с теоретическими оценками сложности алгоритма.

В качестве дальнейших исследований можно предложить оптимизацию алгоритма с точки зрения рассмотрения ограниченного кол-ва масок (так как в процессе полного перебора рассматриваются маски, задающие одинаковое разбиение; таким образом, маски 1 1 1 2 2 2 и 2 2 2 1 1 1 задают одинаковое разбиение: первые три элемента в одном кластере, три других – в другом). Также возможна переработка алгоритма: вместо рассмотрения масок, их можно «строить» исходя из определенных условий, что снизит расходы по времени.

1. Приложения

ПРИЛОЖЕНИЕ А

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 6 - первая часть исходного кода

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

Автоматически созданное описание

Рисунок 7 - вторая часть исходного кода