ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт по лабораторной работе № 5

«Сортировки»

Выполнил работу

Вагин Арсений

Академическая группа №J3112

Принято

Должность, звание Фамилия Имя преподавателя

Санкт-Петербург

2024

**1. Введение**

Цель работы:

Изучить три алгоритма сортировки (вставками, Timsort и ведрами), провести их анализ и сравнение с точки зрения времени выполнения, использования памяти и асимптотической сложности.

Задачи:

Реализовать алгоритмы сортировки.

Провести замеры времени выполнения алгоритмов на массивах различных размеров.

Провести анализ использования памяти.

Построить графики зависимости времени выполнения от размера массива.

Сформулировать выводы на основе проведённых экспериментов.

**2. Теоретическая подготовка**

**Сортировка вставками**

Принцип работы:

Алгоритм проходит по массиву, вставляя каждый элемент на правильное место в отсортированной части массива.

Асимптотика:

1. Лучший случай: O(n), если массив уже отсортирован.
2. Худший случай: O(n2), если массив отсортирован в обратном порядке.
3. Средний случай: O(n2), если массив хаотичен.

Подсчёт памяти:

Временная сложность по памяти: O(1).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1. Код сортировки вставками

**Timsort**

Принцип работы:

Гибридный алгоритм, сочетающий сортировку вставками и сортировку слиянием. Работает с "рунами" — небольшими отсортированными подмассивами.

Асимптотика:

1. Лучший случай: O(n), если массив уже отсортирован.
2. Худший случай: O(n logn).
3. Средний случай: O(n logn).

Подсчёт памяти:

Временная сложность по памяти: O(n), так как для слияния используется дополнительный буфер.

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 2. Код сортировки Timsort

**Сортировка ведрами**

Принцип работы:

Массив разбивается на "ведра" в зависимости от диапазона значений. В каждом ведре элементы сортируются, а затем объединяются в итоговый массив.

Асимптотика:

1. Лучший случай: O(n+k), где k — количество ведёр, если элементы равномерно распределены.
2. Худший случай: O(n2), если все элементы попали в одно ведро.
3. Средний случай: O(n+k).

Подсчёт памяти:

Временная сложность по памяти: O(n+k).

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 3. Код сортировки ведрами

**3. Реализация**

Этапы выполнения:

1. Реализованы алгоритмы сортировки.
2. Проведён анализ на массивах различных размеров (от 1,000 до 1,000,000 элементов).
3. Построены графики времени выполнения и произведён анализ использования памяти.

Изображение выглядит как текст, линия, График, диаграмма

Автоматически созданное описание

Рисунок 4. График сравнения времени работы сортировок

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описание

Рисунок 5. Box Plot. Время работы сортировок при 1е4 элементах

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описание

Рисунок 6. Box Plot. Время работы сортировок при 1е5 элементах

**4. Заключение**

**Выводы**

1. Timsort стабильно работает быстрее других алгоритмов благодаря сочетанию методов.
2. Сортировка вставками эффективна на маленьких массивах, но сильно замедляется на больших.
3. Сортировка ведрами хорошо работает при равномерном распределении данных, но не подходит для хаотичных массивов.
4. Результаты экспериментов согласуются с теоретической асимптотикой.
5. Выбросы наблюдались у ведерной сортировки из-за несбалансированного распределения элементов.

**Применимость алгоритмов:**

* 1. Сортировка вставками: полезна для небольших массивов.
  2. Timsort: универсален и эффективен на любых данных.
  3. Сортировка ведрами: подходит для равномерно распределённых данных.