Лабораторная работа по float

Моргулёв Илья

Октябрь 10, 2023

1 Введение.

1. **Цель работы.** Изучение скорости работы алгоритмов и сортировок с различной асимптотикой, сравнение теоретических сведений с практическими результатами. Изучение различных степеней оптимизации.

2. **Задачи.** Задачи взяты с одноимённой лабораторной работы.

3. **Сторонние функции**
   Время

```
#include <chrono>
double get_time() {
return std::chrono::duration_cast<std::chrono::microseconds>
(std::chrono::steady_clock::now().time_since_epoch()).count();
}
```

   Рандомайзер

```
#include <random>
int rand_uns(int min, int max) {
unsigned seed = std::chrono::steady_clock::now().time_since_epoch().count();
static std::default_random_engine e(seed);
std::uniform_int_distribution<int> d(min, max);
return d(e);
}
```

2 Ход работы.

1^0 **Пузырёк и его товарищи.** Задача: рассмотреть несколько сортировок с квадратичной зависимостью, построить графики зависимости времени от длинны массива, доказать, что зависимость квадратична. Используемые сортировки: Пузырёк, Шейкер, Вставка.

(a) **Пузырёк (Bubble Sort)**
Метод: сравнение двух соседних членов.

(b) **Шейкер (Shaker Sort)**
Метод: основан на пузырьке, ключевым отличием яыляется одновременное рассмотрение массива как слева-направо так и справа-налево.

(c) **Вставка(Insertion Sort)**
Метод: на каждом шаге алгоритма мы выбираем элемент массива и находим ему место для вставки. Условно говоря, двигаем отсортированную подпоследовательность в массиве до тех пор пока не отсортируем весь массив.

Построим графики зависимости времени от количества членов в массиве для всех трех типов сортировок.

Докажем, что зависимость квадратична. Построим графики в логарифмических осях.

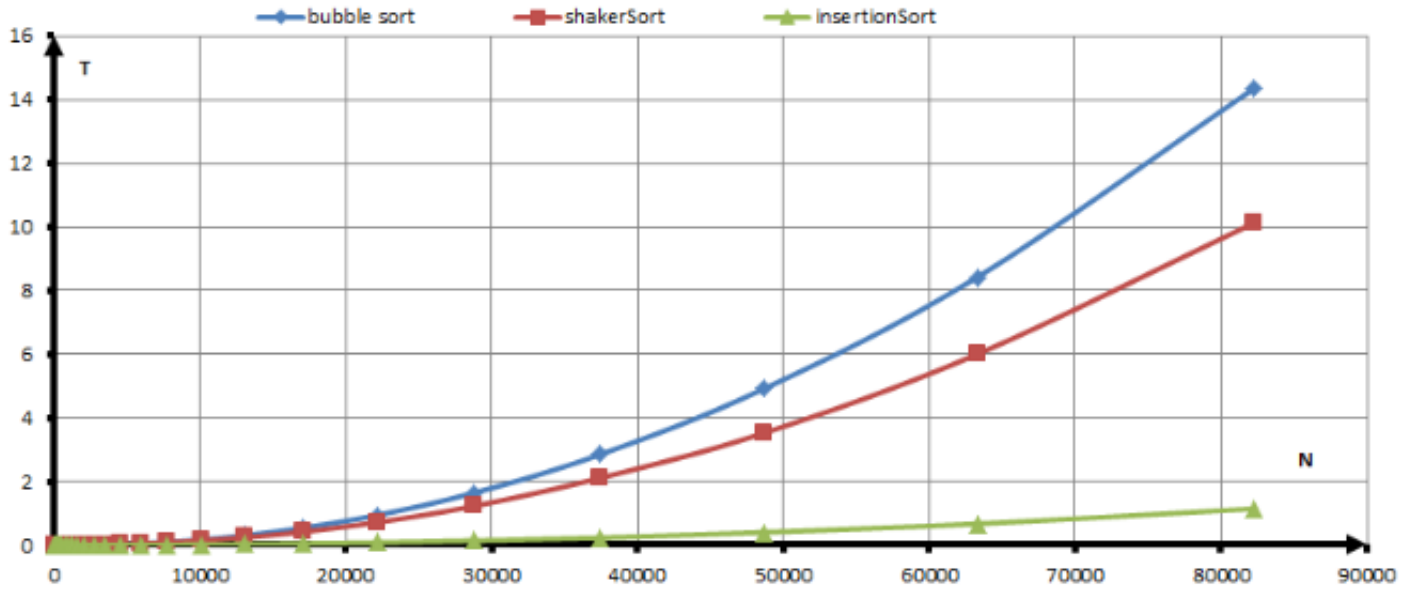

Рис. 1: График зависимости T(N)

$$T = \alpha N^2 \tag{1}$$

$$\log T = \log \alpha N^2 = \log \alpha + \log N^2 = const + 2 \log N \tag{2}$$

$$\log T = C + 2 \log N \tag{3}$$

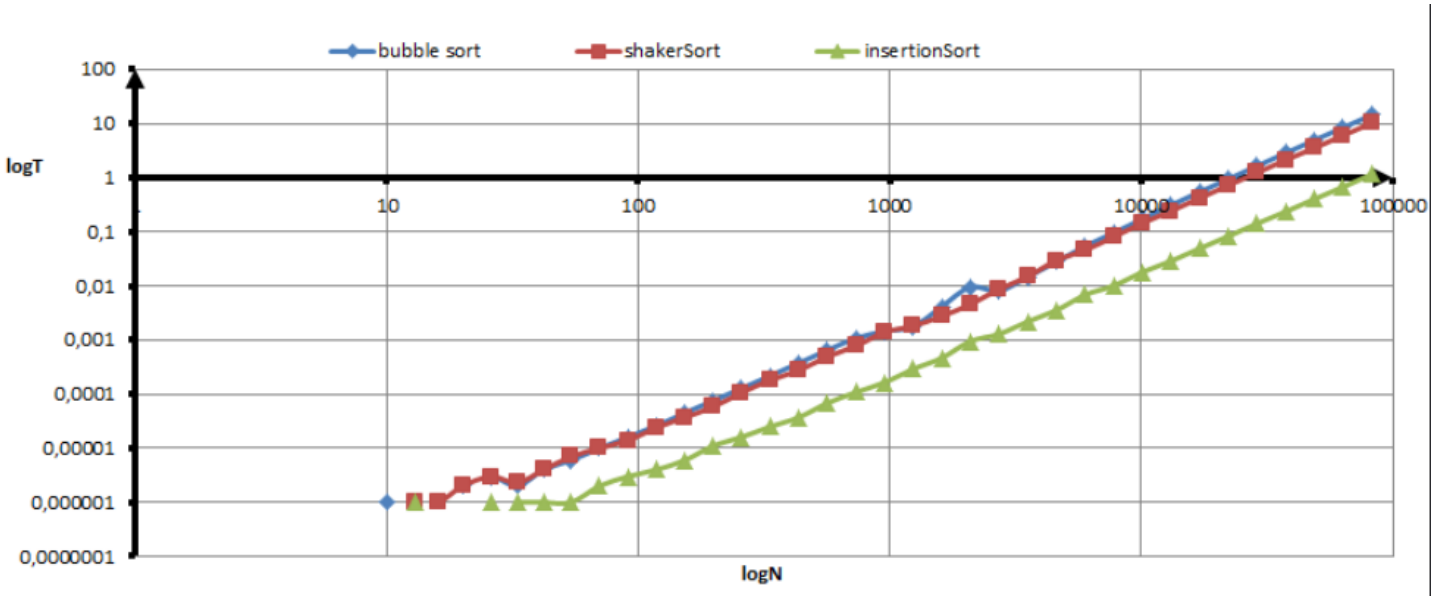

Рис. 2: График зависимости $\log(\log N)$

Мы доказали, что сортировки, перечисленные выше действительно работают за $O(N^2)$

1. **Пузырёк, но быстрее. Оптимизация**
Есть несколько различных вариантов оптимизации процессов:
O_0 - самые простые и примитивные оптимизации
O_1 - более сильные оптимизации
O_2 - оптимизировать всё, что можно, но только проверенные и надёжные оптимизации
O_3 - жёсткая и насильственная оптимизация, применяются экспериментальные методы
Исследовать оптимизацию будем на Пузырьке. Ниже приведён график для зависимости времени от длины массива для разных типов оптимизации.
На графике заметно, что O_0 заметно медленнее.

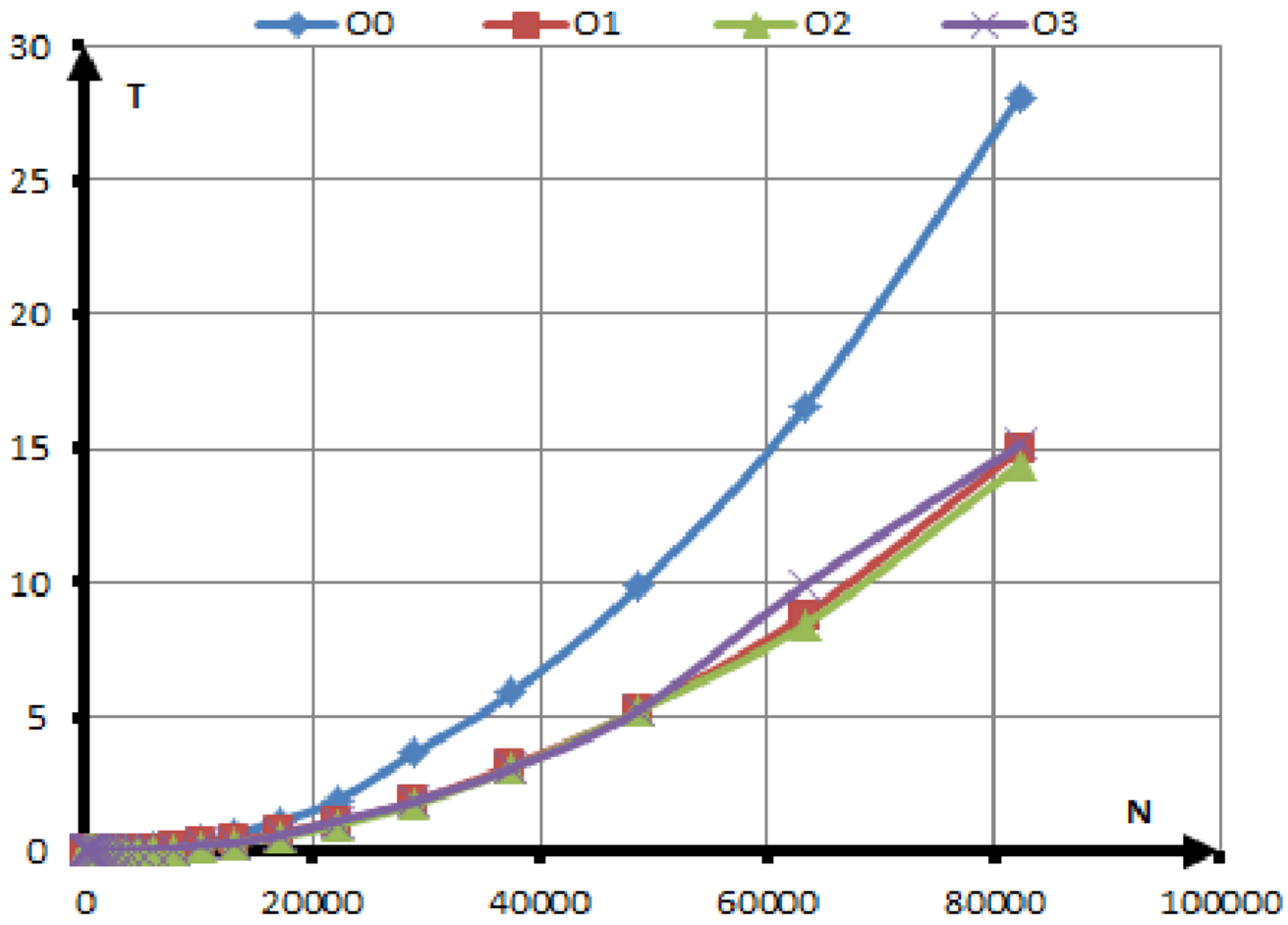

Рис. 3: График зависимости T(N)

2 **А теперь настоящие быстрые сортировки.**
Ниже приведены сортировки, работающие за $O(N \log N)$, и кратко описаны алгоритмы их работы.

(a) **Метод Слияние (Merge Sort)**
Разделяем массив на две половины и так делим, пока не получим подмассивы длины 1, потом каждые две половины сортируем между собой и объединяем. Повторяем до тех пор, пока весь массив не будет отсортирован.

(b) **Быстрая сортировка (Quick Sort)**
Выбираем в массиве некоторый элемент, который называется разрешающим. Затем он помещается в то место массива, где ему полагается быть после упорядочивания всех элементов. В процессе поиска подходящего места для разрешающего элемента производятся перестановки элементов так, что слева от них находятся элементы меньше разрешающего, а справа - больше.

(c) **Комбо Сортировка (Comb Sort)**
Изначально берёт большое расстояние между сравниваемыми элементами, в процессе сортировки сужая это расстояние. (Расчёска:))
Отразим на графике время работы для этих сортировок.

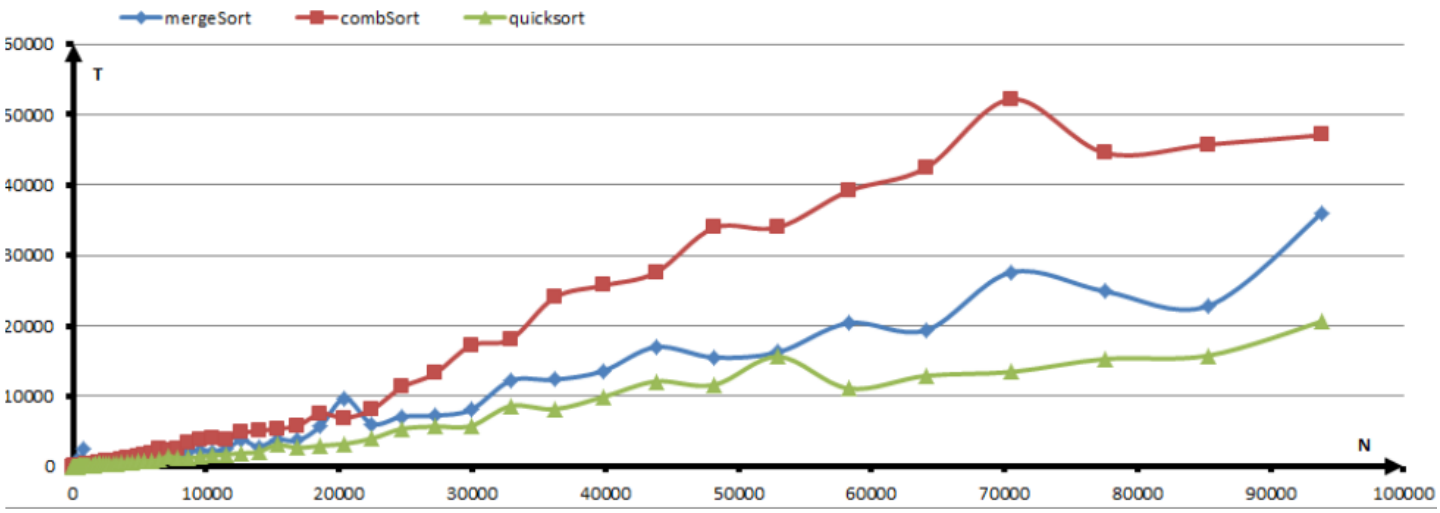

Рис. 4: График зависимости T(N)

Докажем, что эти сортировки действительно работают за $O(N \log N)$

$$T = \alpha N \log N \tag{4}$$

$$\alpha = \frac{T}{N \log N} \approx const \tag{5}$$

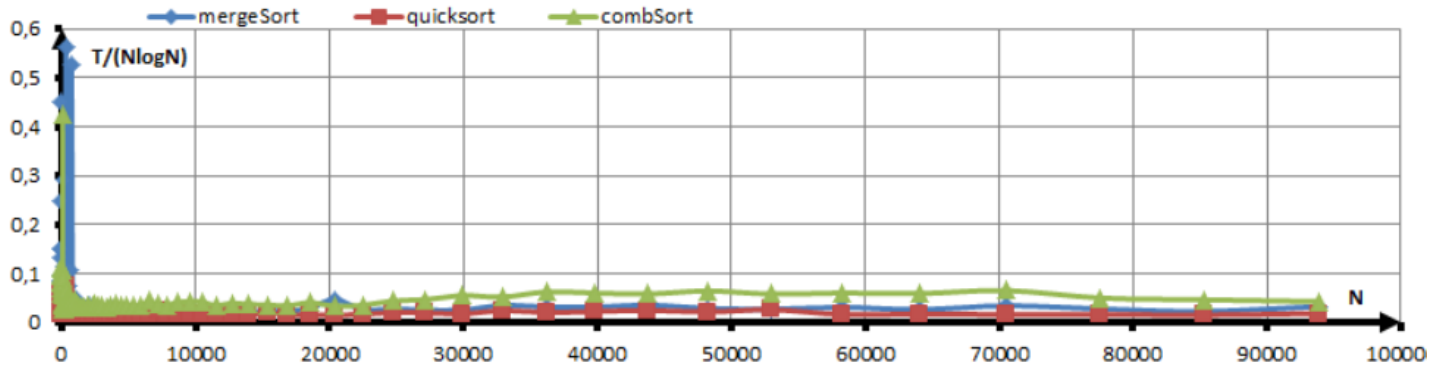

Рис. 5: График зависимости $\alpha(N)$

Мы доказали, что эти сортировки действительно работают за $O(N \log N)$

3. $O(N^2)$ **vs** $O(N \log N)$
   Отразим на одном графике зависимоти от времени для всех ранее описанных сортировок.

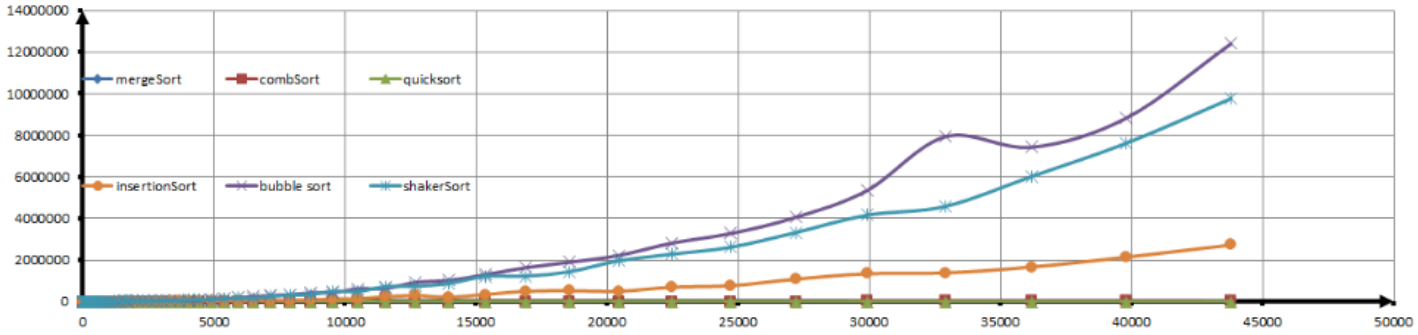

Рис. 6: График зависимости T(N)

Наглядно видно, что квадратичные сортировки работают кратно дольше.

4. **Зависимость от начальных данных**
   Рассмотрим, что будет происходить с зависимостями, если сортировки работают с уже отсортированным массивом (BEST), массиве рандомных чисел (GOOD) и массиве, отсортированном в обратном порядке (BAD).

   (a) **Пузырёк**

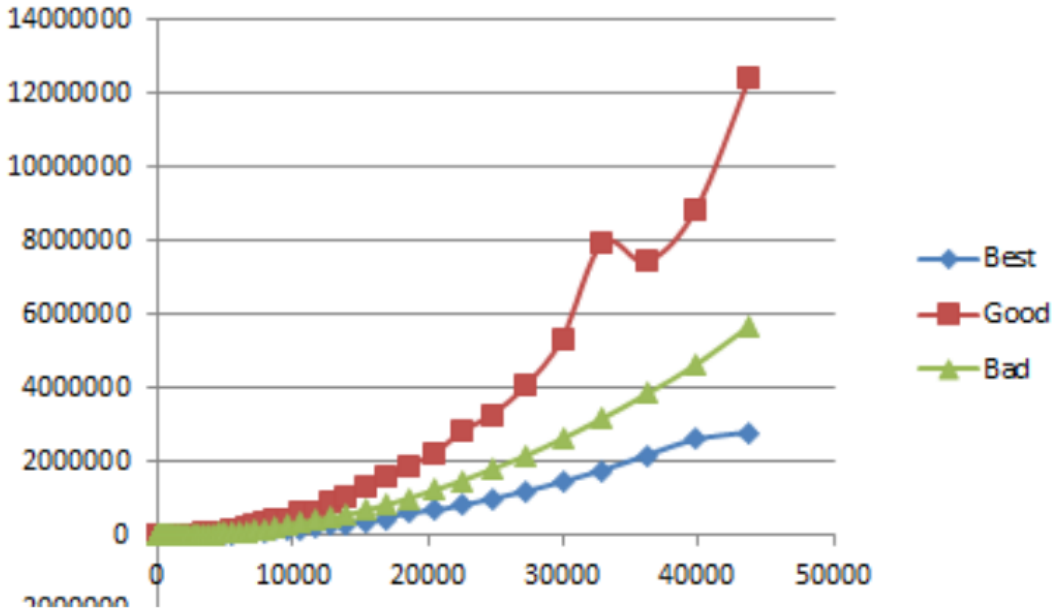

Рис. 7: Три случая для Пузырька

(b) **Шейкер**

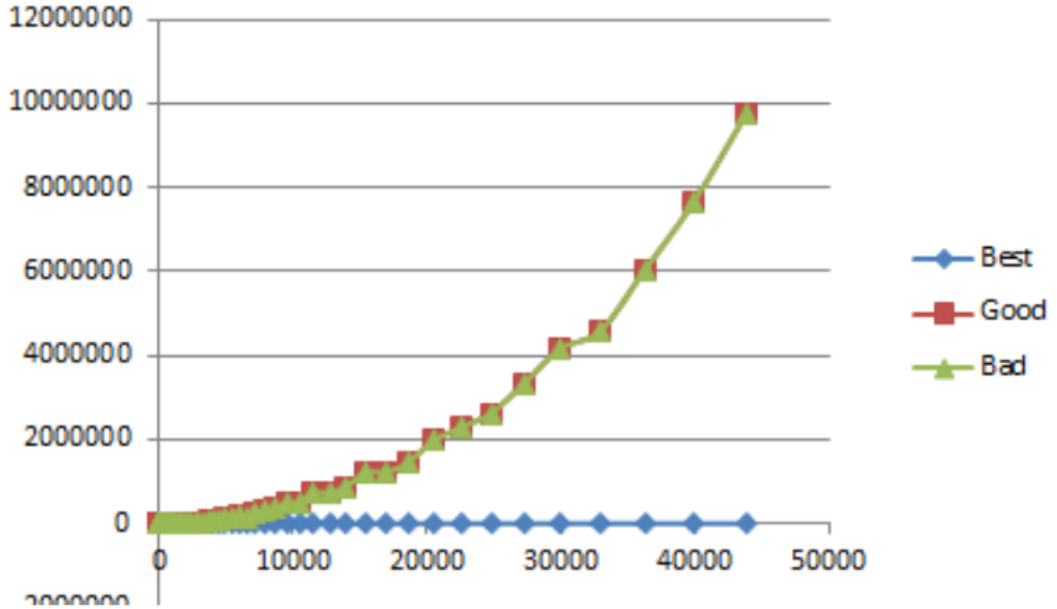

Рис. 8: Три случая для Шейкера

(c) **Вставка**

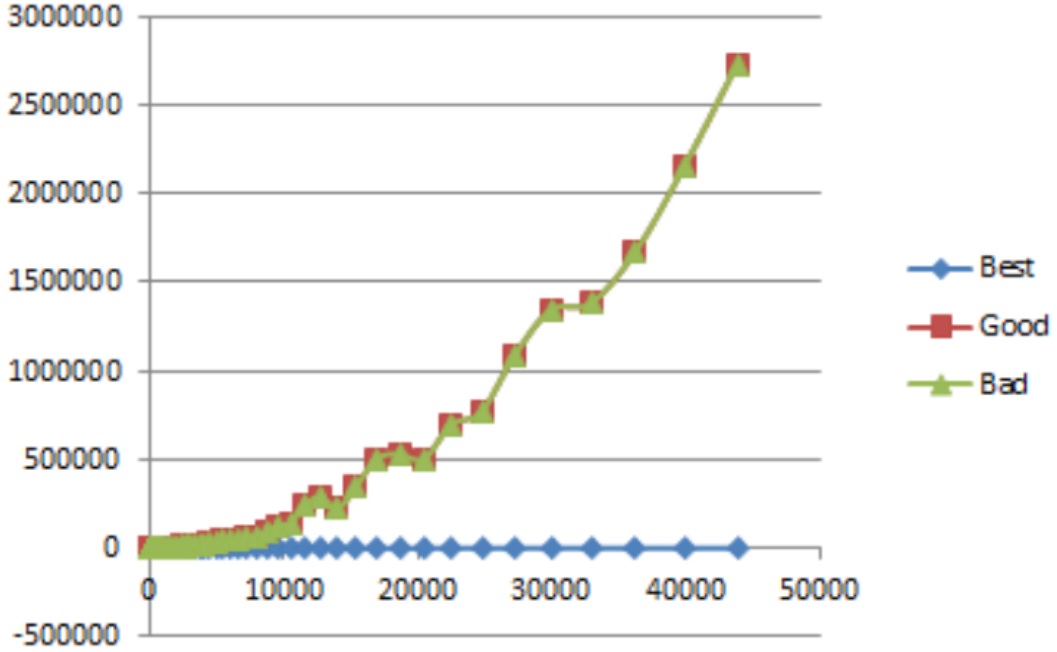

Рис. 9: Три случая для Вставка

(d) **Выбор**

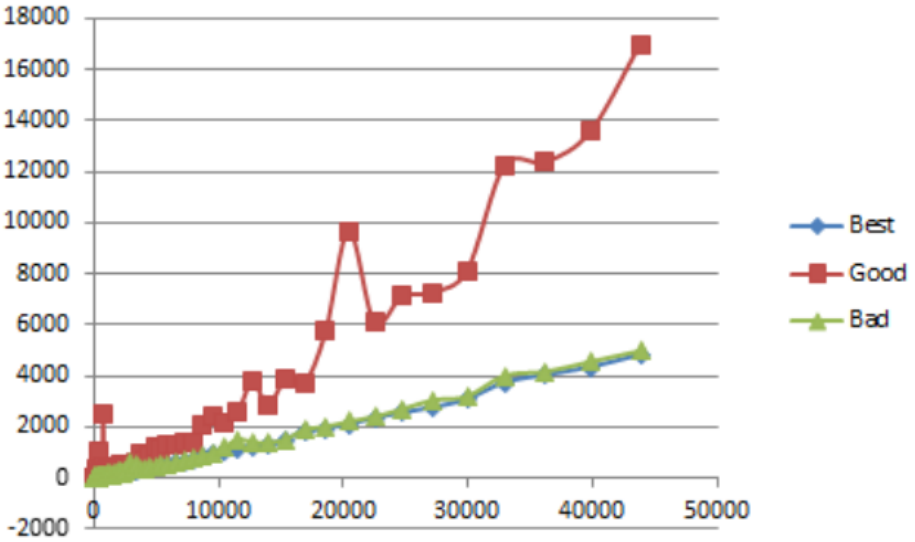

Рис. 10: Три случая для Выбора

(e) **Расчёска**

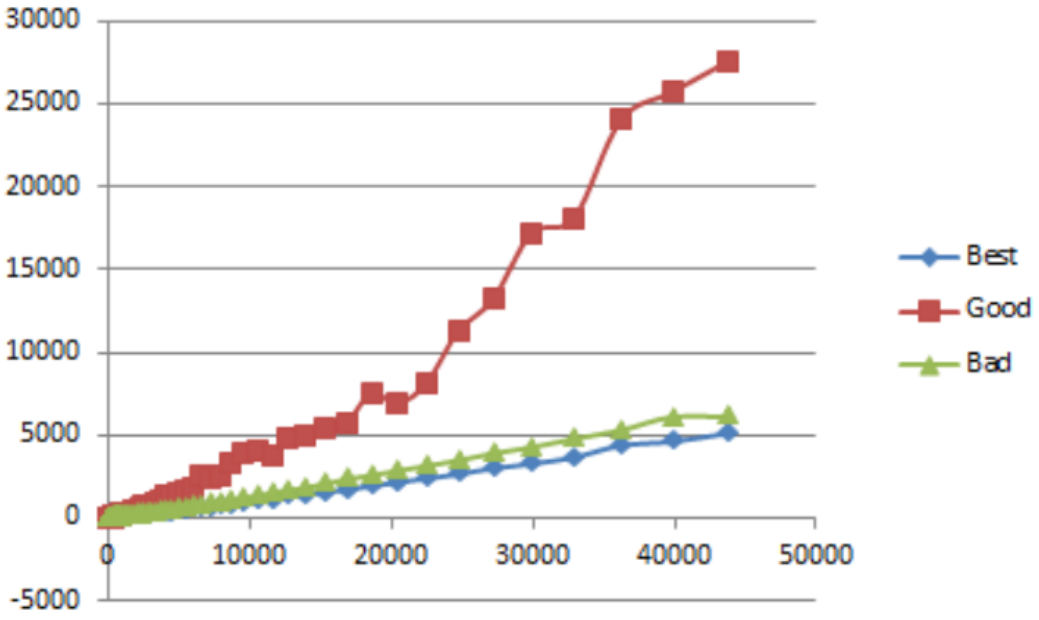

Рис. 11: Три случая для Расчёски

(f) **Быстрая**

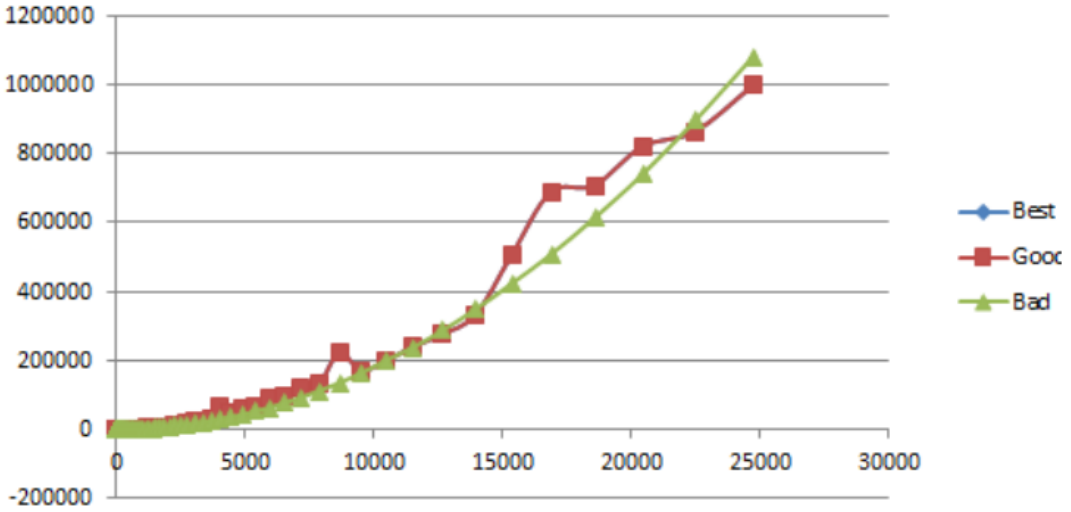

Рис. 12: Три случая для Быстрой

4. **Сравнение всех сортировок.**
   Изобразим на одном графике последние шесть зависимостей (да, безумие, 18 зависимостей на одном графике))

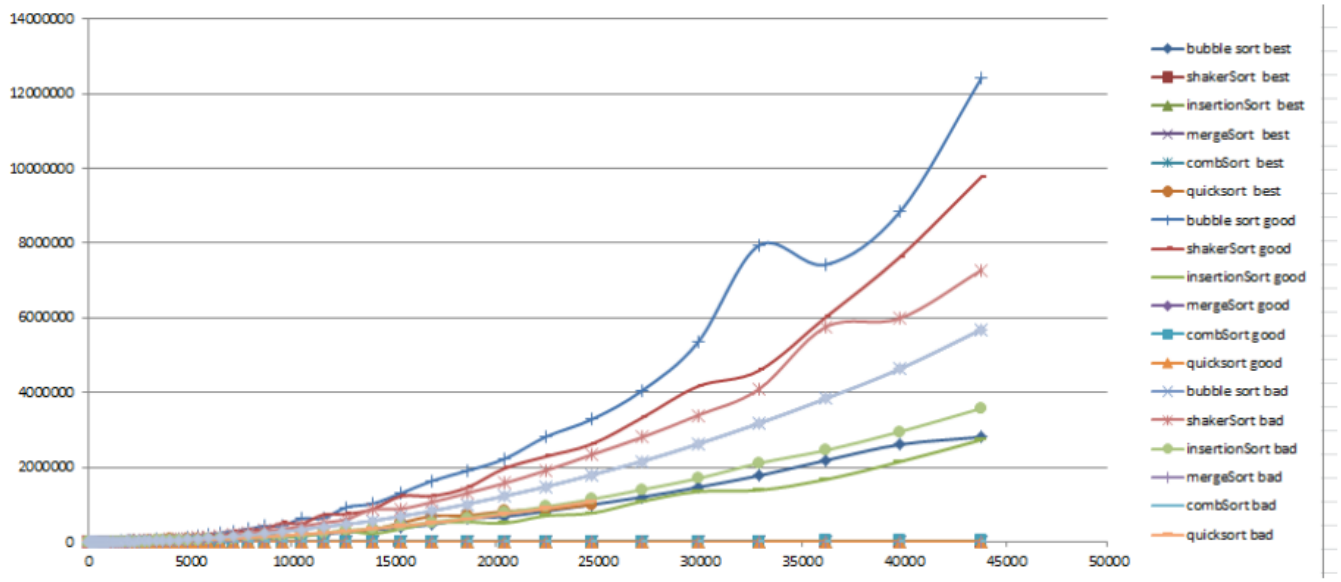

Рис. 13: Общее

3 Вывод.

Мы рассмотрели различные типы сортировок и на деле убедились в верных оценках асимптотики их зависимостей.