Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Российский химико-технологический университет имени Д.И. Менделеева»
Кафедра информационных компьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

Выполнил студент группы.....КС-36 Жижин Алексей Павлович

Ссылка на репозиторий: https://github.com/MUCTR-IKT-CPP/APZhizhin_36_2SEM

Приняли: .....Пысин Максим Дмитриевич
.....Краснов Дмитрий Олегович

Дата сдачи: 21.02.2022

Оглавление

Описание задачи.

Необходимо написать алгоритм быстрой сортировки

Описание метода/модели.

QuickSort является существенно улучшенным вариантом алгоритма сортировки с помощью прямого обмена (его варианты известны как «Пузырьковая сортировка» и «Шейкерная сортировка»), известного в том числе своей низкой эффективностью. Принципиальное отличие состоит в том, что в первую очередь производятся перестановки на наибольшем возможном расстоянии и после каждого прохода элементы делятся на две независимые группы (таким образом улучшение самого неэффективного прямого метода сортировки дало в результате один из наиболее эффективных улучшенных методов).

Общая идея алгоритма состоит в следующем:

- Выбрать из массива элемент, называемый опорным. Это может быть любой из элементов массива. От выбора опорного элемента не зависит корректность алгоритма, но в отдельных случаях может сильно зависеть его эффективность
- Сравнить все остальные элементы с опорным и переставить их в массиве так, чтобы разбить массив на три непрерывных отрезка, следующих друг за другом: «элементы меньшие опорного», «равные» и «большие»
- Для отрезков «меньших» и «больших» значений выполнить рекурсивно ту же последовательность операций, если длина отрезка больше единицы.

На практике массив обычно делят не на три, а на две части: например, «меньшие опорного» и «равные и большие»; такой подход в общем случае эффективнее, так как упрощает алгоритм разделения (см. ниже).

Хоар разработал этот метод применительно к машинному переводу; словарь хранился на магнитной ленте, и сортировка слов обрабатываемого текста позволяла получить их переводы за один прогон ленты, без перемотки её назад. Алгоритм был придуман Хоаром во время его пребывания в Советском Союзе, где он обучался в Московском университете компьютерному переводу и занимался разработкой русско-английского разговорника.

Алгоритм состоит из трёх шагов:

1. Выбрать элемент из массива. Назовём его опорным.
2. Разбиение: перераспределение элементов в массиве таким образом, что элементы, меньшие опорного, помещаются перед ним, а большие или равные - после.

3. Рекурсивно применить первые два шага к двум подмассивам слева и справа от опорного элемента. Рекурсия не применяется к массиву, в котором только один элемент или отсутствуют элементы.

Выполнение задачи.

Алгоритм написан на языке программирования Java. Были проведены тесты на скорость работы алгоритма(Рис. 1), на количество дополнительной памяти(Рис. 2), на количество рекурсий(Рис. 3).

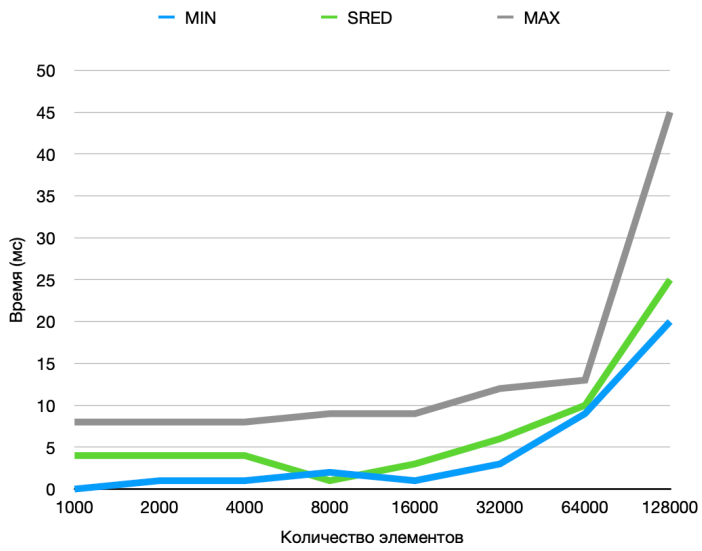

Рис. 1

На рис. 1 алгоритм показывает отличные результаты работы. Не смотря на то, что я провел 160 тестов, работа программы заняла буквально 1 секунд, что говорит о том, что сортировка правда быстрая, что касается тестов на 128000 элементов, сортировка показывает время работы меньше 20 милисекунды, что довольно хорошо

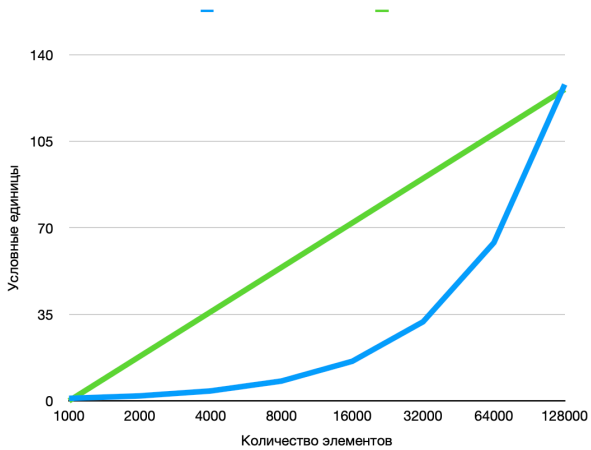

Рис. 2

На рис. 2 видно квадратичную зависимость увеличение временной памяти от количества элементов. Именно так выглядит зависимость, если нашу память не очищать. Java имеет специальный инструмент по очистке памяти, который очищает память из под объекта, который давно не использовался.

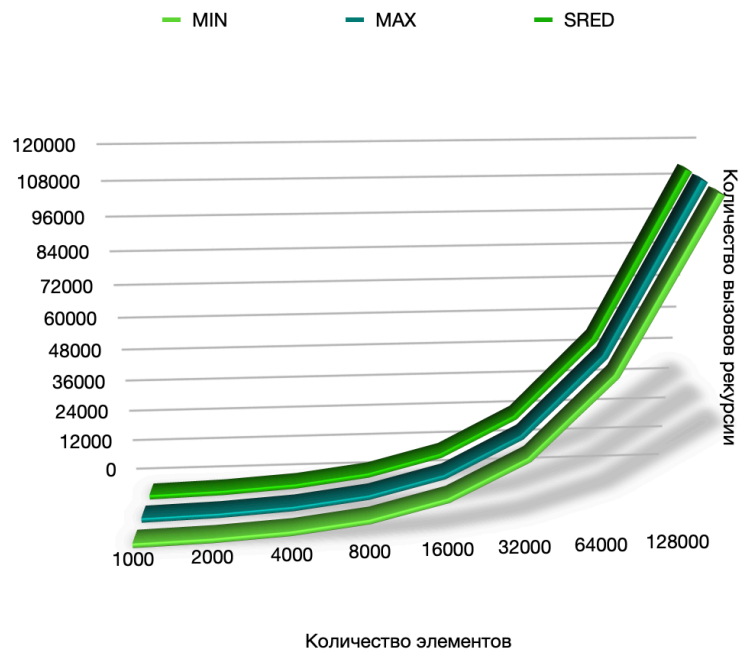

Рис. 3

На рис. 3 приведена зависимость количества рекурсий от количества элементов. На графике мы видим, что вызов рекурсии растет квадратично количествам элементов. Рекурсия имеет как свои плюсы, так и минусы. К плюсам относиться: читабельность кода, простота написания кода. К минусам: выделение памяти. Но в целом рекурсия- важный момент, программирования, так как она позволяет тратить меньше времени на написание и анализ кода.

Заключение.

Я считаю, что алгоритм быстрой сортировки- отличный алгоритм. Он справился со всеми тестами в считанное мгновение, показал отличный результат на разных наборах данных. Имеет одну из лучших алгоритмических сложностей даже в худшем случае. Единственное какой минус он имеет- многократное использование рекурсивного подхода, который негативно складывается на ресурсах компьютера. Но у каждого алгоритма есть свои плюсы и минус и я считаю, что в этом случае, лучше пожертвовать ресурсами компьютера и получить плюс во времени работы.