Сахибназарова Виктория Бахтиёровна
студентка
Сайгак Кристина Олеговна
студентка
ФГАОУ ВО «Самарский государственный
аэрокосмический университет
им. академика С.П. Королёва (НИУ)»
г. Самара, Самарская область

РАЗРАБОТКА И АНАЛИЗ АЛГОРИТМА СОРТИРОВКИ НА ОСНОВЕ ИСПОЛЬЗОВАНИЯ БИНАРНЫХ ДЕРЕВЬЕВ

***Аннотация**: в данной работе рассмотрены основные понятия, связанные с деревьями и бинарными деревьями, составлен и проанализирован алгоритм сортировки с помощью бинарных деревьев, а также проведено сравнение эффективности разработанного алгоритма с уже известными.*

***Ключевые слова**: бинарные деревья, алгоритм сортировки, анализ сложности.*

Бинарное дерево поиска – это достаточно базовая, но в то же время интересная структура данных, у которой, к тому же, существует большое количество модификаций и вариаций, а также применений на практике.

В теории графов дерево представляет собой ориентированный или неориентированный граф, не содержащий циклов. То есть для любой вершины существует один и только один способ добраться до любой другой вершины. В программировании широко используются как бинарные деревья, в которых число исходящих дуг не превосходит двух, и N-арные деревья (с произвольным количеством исходящих ребер).

Сортировка с помощью двоичного дерева – универсальный алгоритм сортировки, заключающийся в построении двоичного дерева поиска по ключам массива (списка), с последующей сборкой результирующего массива путём обхода узлов построенного дерева в необходимом порядке следования ключей. То есть

если мы будем выполнять операцию обхода дерев, то, записывая все встречающиеся элементы в массив, получим упорядоченное в порядке возрастания множество.

Мы разработали собственный Бинарный алгоритм сортировки массивов на основе бинарных деревьев, который основан на том, что для каждой вершины, начиная с корня, используется правило «влево уходит старший сын, вправо – младший брат». Алгоритм Бинарной сортировки (по возрастанию):

Шаг 1. Преобразование к бинарному дереву: Исходную последовательность данных представляем в виде бинарного дерева таким образом, что для каждого элемента в левое поддерево уходят числа, большие либо равные предыдущему числу последовательности, а в правое – меньшие. В результате получаем структуру в виде одной ветки.

Шаг 2. Сравнение: Начиная с самого нижнего листа, мы движемся вверх до тех пор, пока не обнаружим, что следующая вершина имеет большее значение (если данное условие не выполняется, значит, последовательность отсортирована).

Шаг 3. Выделение пустой вершины: К найденной вершине, значение которой больше значения потомка, навешиваем вторую пустую вершину (значение не определено).

Шаг 4. Сдвиг: Вершина с большим значением сдвигается на пустое место (в левое поддерево), правое поддерево смещается верх на одну позицию. В результате, крайняя дочерняя вершина оказывается с пустым значением. Освободившаяся пустая вершина удаляется.

Шаг 5. Приведение к виду бинарного дерева: Представляем получившееся упорядоченное дерево в виде ветки.

На рисунке 1 показано сравнение времени выполнения данного алгоритма сортировки с пузырьковой и древесной алгоритмами. Как известно, сложность алгоритма прямо зависит от длины последовательности. Так для простейшей сортировки пузырьком эта зависимость квадратичная O(n^2), где n – количество

элементов сортируемого массива, что весьма неэффективно. Общее быстродействие метода древесной сортировки составляет O(n$\log n$), основным недостатком является требование к памяти под дерево.

Что касается приведенного созданного нами алгоритма, сложность данного алгоритма составляет O($n^2 \log n$) в среднем случае, где шаги преобразования массива данных к бинарному дереву и обратно составляет O(n), а операции вставки и удаления элементов в среднем случае имеют сложность O($\log n$), а в случае разбалансированного дерева O(n), что может привести к сложности всего алгоритма Бинарной сортировки порядка O(n^3).

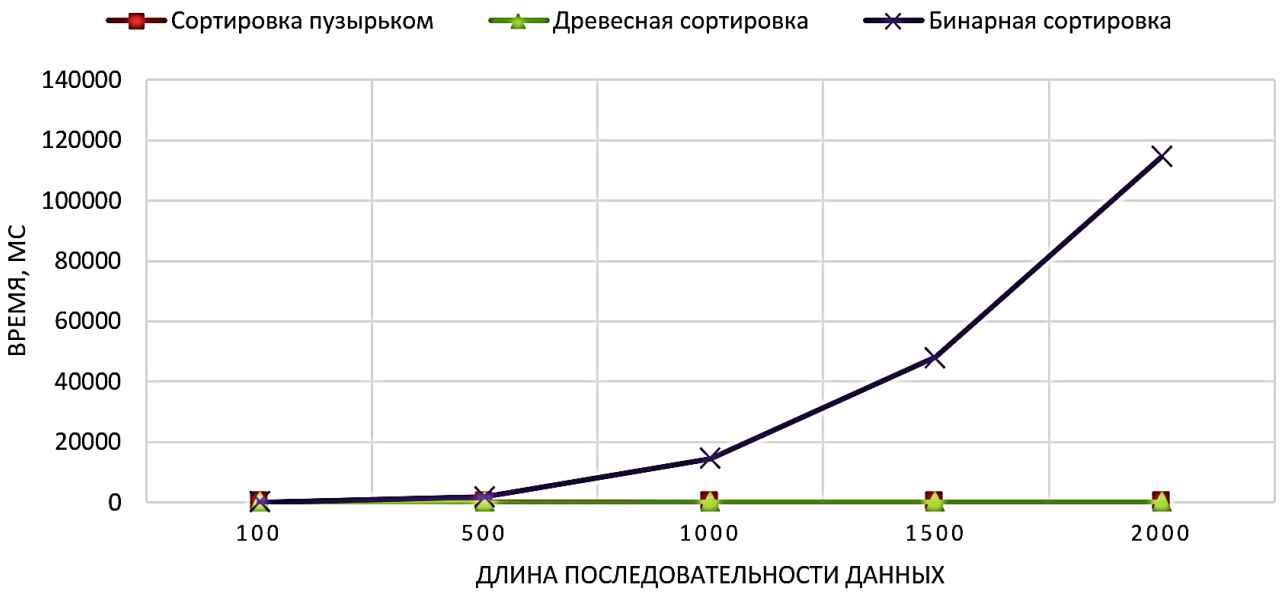

Рис. 1. График зависимости времени сортировки от длины последовательности

Недостатком являются также большие требования к памяти. Очевидно, нужно n места под ключи и, кроме того, память на 2 указателя для каждого из них.

Таким образом, такой алгоритм нельзя отнести к группе быстрых, т.к. при большом количестве входных данных его использование не является оптимальным по времени и расходам памяти. Значит, существующий алгоритм древесной сортировки является более эффективным.

Список литературы

1. Уилсон В. Введение в теорию графов. – М.: Мир, 1977. – С. 57–72.

2. Практикум по программированию на языке Паскаль: Массивы, строки, файлы, рекурсия, линейные динамические структуры, бинарные деревья. – 6-е изд., перераб. и доп. – Ростов н/Д: ЦВВР, 2008. – 227 с.

3. Бинарные деревья: задачи, решения, указания. – Банк компьютерных изданий ЮФУ, 2009. – 71 с.