УДК 004.424.(5)

ЧИСЛЕННЫЙ АНАЛИЗ МЕТОДОВ СОРТИРОВКИ

Сотанин С. В.
Научный руководитель – старший преподаватель Рыбас Н.А.

Сибирский федеральный университет

В общем случае сортировку следует понимать как процесс перегруппировки заданного множества объектов в определенном порядке.

В информатике, по-видимому, нет более глубоко исследованных задач, чем задачи сортировки и поиска. Алгоритмы сортировки и поиска чрезвычайно широко распространены практически во всех задачах обработки информации. При этом они настолько тесно связаны друг с другом, что образуют отдельный класс алгоритмов. Действительно, алгоритмы сортировки, как правило, применяются с целью последующего более быстрого поиска. Трудно себе представить, как бы мы пользовались словарями, если бы слова (или словарные статьи) в них не были бы упорядочены по алфавиту.

Алгоритмы сортировки имеют большое практическое применение. Их можно встретить там, где речь идет об обработке и хранении больших объемов информации. Некоторые задачи обработки данных решаются проще, если данные заранее упорядочить.

Рассмотрение алгоритмов сортировки естественно начать с самых простых, хотя и одновременно самых неэкономичных методов.

Методы сортировки массивов можно разбить на 4 основных класса в зависимости от лежащего в их основе приема:

1. Сортировка обменом.
2. Сортировка выбором.
3. Сортировка включением.
4. Сортировка слиянием.

На сегодняшний день существует множество различных алгоритмов сортировки. Для оценки эффективности работы того или иного алгоритма используются следующие критерии:

- Скорость работы алгоритма сортировки
- Время работы в лучшем и худшем случаях
- Поведение алгоритма сортировки

Различные сортировки массивов отличаются по быстродействию. Существуют простые методы сортировок, которые требуют порядка n^2 сравнений, где n – количество элементов массива и быстрые сортировки, которые требуют порядка $n*ln(n)$ сравнений. Простые методы удобны для объяснения принципов сортировок, т.к. имеют простые и короткие алгоритмы. Усложненные методы требуют меньшего числа операций, но сами операции более сложные, поэтому для небольших массивов простые методы более эффективны.

Самый известный алгоритм – **пузырьковая сортировка**. Его популярность объясняется интересным названием и простотой самого алгоритма.

Алгоритм состоит в повторяющихся проходах по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов. Проходы по массиву повторяются до тех пор, пока на очередном проходе окажется, что обмены больше не нужны, это означает –

массив отсортирован. В результате элемент, стоящий не на своём месте, как бы "всплывает" до нужной позиции.

Но пузырьковая сортировка имеет существенный недостаток: один «легкий пузырек», расположенный внизу массива, «всплывает» на свое место за один проход, а «тяжелый пузырек», расположенный наверху, «опускается» на свое место только на один шаг за один проход.

Сортировка методом выбора. Это наиболее естественный алгоритм упорядочивания. При данной сортировке из массива выбирается элемент с наименьшим значением и обменивается с первым элементом. Затем из оставшихся *n-1* элементов снова выбирается элемент с наименьшим значением и обменивается со вторым элементом, и т.д.

Сортировка включением. При сортировке включением сначала упорядочиваются первые два элемента массива. Затем последовательно делается вставка третьего элемента в соответствующее место по отношению к первым двум элементам, затем вставка четвертого элемента в список из трех элементов и т.д. Этот процесс повторяется до тех пор, пока все элементы не будут упорядочены. Метод выбора очередного элемента из исходного массива произволен; может использоваться практически любой алгоритм выбора.

Сортировка слиянием. Слияние означает объединение двух (или более) последовательностей в одну-единственную упорядоченную последовательность с помощью повторяющегося выбора из доступных в данный момент элементов. Слияние намного проще сортировки, и его используют как вспомогательную операцию в более сложных процессах сортировки последовательностей. Одна из сортировок на основе простого слияния называется прямым слиянием. Она выполняется следующим образом:

1. Последовательность *a* разбивается на две половины *b* и *c*.

2. Части *b* и *c* сливаются, при этом одиночные элементы образуют упорядоченные пары.

3. Полученная последовательность под именем a вновь обрабатывается, как указано в пунктах 1 и 2; при этом упорядоченные пары переходят в такие же четвёрки.

4. Повторяя предыдущие шаги, сливаем четвёрки в восьмёрки и т.д., каждый раз «удваивая» длину слитых последовательностей до тех пор, пока не будет упорядочена вся последовательность.

Все вышеперечисленные сортировки являются сортировками порядка n^2, что делает их слишком медленными и неэффективными на больших объёмах данных.

Сортировка Шелла. Общий метод, который использует сортировку включением, применяет принцип уменьшения расстояния между сравниваемыми элементами. Сначала сортируются все элементы, смещенные друг от друга на позиции. Затем сортируются все элементы, которые смещены на две позиции. И, наконец, упорядочиваются все соседние элементы.

Сортировка методом Хоара или быстрая сортировка. Алгоритм быстрой сортировки использует обменный метод сортировки. В его основе принцип разбиения. Сначала выбирается некоторое значение в качестве «основы», и затем весь массив разбивается на две части. Одну часть составляют все элементы, равные или большие «основы», а другую часть составляют все элементы меньшего значения, по которому делается разбиение. Этот процесс продолжается для оставшихся частей до тех пор, пока весь массив не будет отсортирован.

Быстрая сортировка является наиболее эффективным алгоритмом из всех известных методов сортировки, но все усовершенствованные методы имеют один общий недостаток – невысокую скорость работы при малых значениях n.

Заканчивая обзор методов сортировки, я попытаюсь сравнить их эффективность. Для всех прямых (простых) методов сортировки можно дать точные аналитические

формулы (рис. 1). Столбцы таблицы определяют минимальное, усреднённое и максимальное по всем *n!* перестановкам из *n* элементов значения, *C* – количество сравнений, а *M* – обменов.

	МИН.	СРЕД.	МАКС.
Метод обмена	$C=(n^2-n)/2$ $M=0$	$C=(n^2-n)/2$ $M=(n^2-n)*0,75$	$C=(n^2-n)/2$ $M=(n^2-n)*1,5$
Метод выбора	$C=(n^2-n)/2$ $M=3*(n-1)$	$C=(n^2-n)/2$ $M=n*(ln(n)+0,57)$	$C=(n^2-n)/2$ $M=n^2/4+3*(n-1)$
Метод включения	$C=(n-1)$ $M=2*(n-1)$	$C=(n^2+n-2)/4$ $M=(n^2-9n-10)/4$	$C=(n^2-n)/2-1$ $M=(n^2-3n-4)/2$

Рис. 1

Анализ модифицированных сортировок достаточно сложен с математической точки зрения. Можно сказать лишь то, что время выполнения сортировки Шелла пропорционально $n^{1.2}$, а для быстрой сортировки число операций сравнения равно $n \cdot log_2 n$, а число операций обмена равно приблизительно $n \cdot log_2 n/6$. Эта зависимость значительно лучше квадратичной зависимости, которой подчиняются рассмотренные ранее алгоритмы сортировки.

Для проверки на практике вышесказанного, мной была написана программа, осуществляющая сортировку массива задаваемой размерности, любым из вышеперечисленных методов, а также создающая таблицы эффективности работы каждого метода. На рисунке 2 приведена укрупнённая блок-схема моей программы:

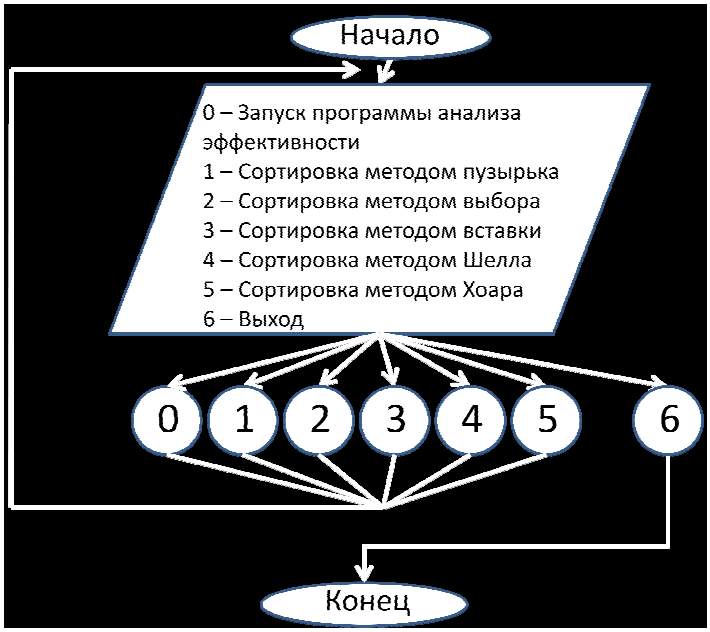


Рис. 2

В работе на была рассмотрена сортировка методом слияния, т.к. она коренным образом отличается от трёх других простых сортировок: в отличие от них, в сортировке слиянием массив сортируется не на месте, а создаётся новый массив по размерности равный исходному, что двукратно увеличивает затраты памяти.

Результат работы программы, точнее, её части – подпрограммы анализа эффективности сортировок, приведен на (рис.3).

В заголовках строк цифрами от 1 до 5 обозначены методы сортировки – пузырьковая, выбором, включением, сортировка Шелла и последняя – быстрая сортировка, а в заголовках столбцов указано количество элементов в сортируемых массивах.

Программа осуществляет сортировку массива задаваемой пользователем размерности методом обмена, выбора, включения, Шелла и быстрой сортировкой. Кроме того, она проводит сравнительный анализ всех алгоритмов сортировки для размерности массива 100, 1000 и 10000 элементов.

После запуска программы выводится текстовое меню с вариантами выбора нужного модуля.

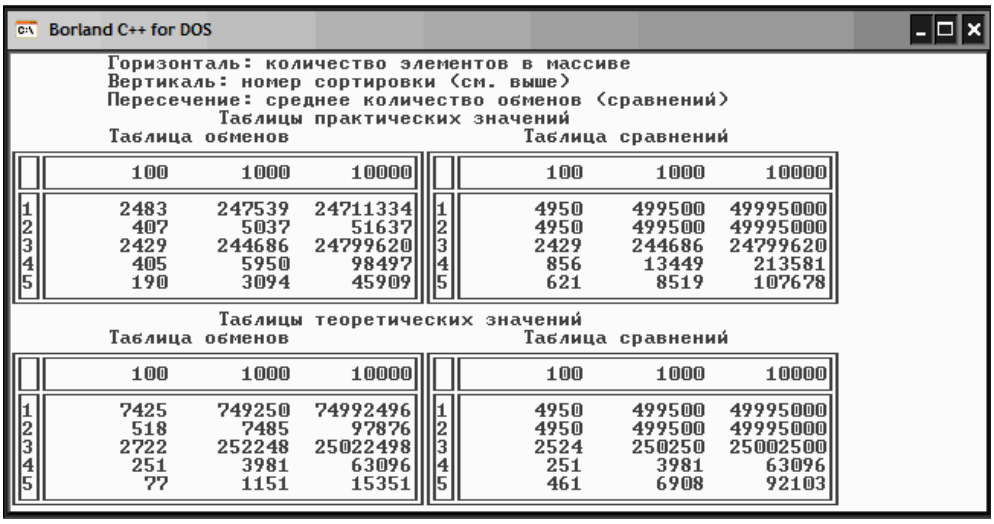
При выборе любого пункта от 1 до 5 программа запросит ввод размерности массива и после ввода размерности запустит соответствующую подпрограмму, выполняющую сортировку массива заданной размерности, заполненного случайными числами. После сортировки программа выведет на экран исходный массив, отсортированный массив, а также количество выполненных операций сравнения и обмена.

После этого вновь будет выведено текстовое меню и программа перейдёт в режим ожидания.

Если выбрать пункт 0 – программа запустит подпрограмму анализа эффективности, которая выполнит каждую из пяти сортировок по 10 раз для массива из 100 элементов и вычислит по результатам 10 запусков среднее значение количества обменов и сравнений. Потом программа выполнит то же самое для массива, состоящего из 1000 элементов, а затем для массива, состоящего из 10000 элементов. Из полученных результатов будут составлены 2 таблицы: таблица обменов и таблица сравнений.

После этого программа рассчитает среднее количество обменов и сравнений для всех пяти алгоритмов по формулам. При расчёте будут использованы такие же размерности массивов, какие использовались при практических расчётах (100, 1000, 10000). Все теоретические данные также будут выведены в виде двух таблиц: таблицы обменов и таблицы сравнений.

По завершении подпрограммы анализа на экран будут выведены четыре таблицы, по которым пользователь сможет проанализировать эффективность каждой из сортировок, т.е. сравнить параметры различных сортировок друг с другом или с теоретическими значениями и сделать выводы о скорости работы каждого из алгоритмов:



	100	1000	10000
1	2483	247539	24711334
2	407	5037	51637
3	2429	244686	24799620
4	405	5950	98497
5	190	3094	45909

	100	1000	10000
1	4950	499500	49995000
2	4950	499500	49995000
3	2429	244686	24799620
4	856	13449	213581
5	621	8519	107678

	100	1000	10000
1	7425	749250	74992496
2	518	7485	97876
3	2722	252248	25022498
4	251	3981	63096
5	77	1151	15351

	100	1000	10000
1	4950	499500	49995000
2	4950	499500	49995000
3	2524	250250	25002500
4	251	3981	63096
5	461	6908	92103

Рис.3

Как видно из приведённых таблиц, некоторые практические значения получились несколько отличными от значений теоретических, но это отличие не очень большое, учитывая размер сортируемого массива. Все простые алгоритмы сортировки обладают очень серьёзным недостатком, а именно, время их выполнения пропорционально квадрату числа элементов. Для больших объёмов данных эти сортировки будут медленными, а начиная с некоторой величины размерности массива, они будут слишком медленными, чтобы их можно было использовать на практике.