1.1 Анализ алгоритмов сортировки

Сортировка простыми обменами, сортировка пузырьком – простой алгоритм сортировки. Для понимания и реализации этот алгоритм — простейший, но эффективен он лишь для небольших массивов. Сложность алгоритма: $O(n^2)$. В то же время метод сортировки обменами лежит в основе некоторых более совершенных алгоритмов, таких как шейкерная сортировка, пирамидальная сортировка и быстрая сортировка.

Сортировка вставками – достаточно простой алгоритм. Основным преимуществом алгоритма сортировки вставками является возможность сортировать массив по мере его получения.

Сортировка выбором – алгоритм сортировки. Может быть как устойчивый, так и неустойчивый. На массиве из n элементов имеет время выполнения в худшем, среднем и лучшем случае $O(n^2)$, предполагая что сравнения делаются за постоянное время.

Быстрая сортировка, сортировка Хоара – один из самых быстрых известных универсальных алгоритмов сортировки массивов: в среднем $O(n \log n)$ обменов при упорядочении n элементов; из-за наличия ряда недостатков на практике обычно используется с некоторыми доработками.

N	select (1)	select (2)	quick	bubble
5	0.00001168	0.00001121	0.00002050	0.00001121
10	0.00002074	0.00002098	0.00003767	0.00002551
50	0.00021791	0.00022483	0.00036836	0.00040555
100	0.00075674	0.00080824	0.00140834	0.00161719
150	0.00161386	0.00177860	0.00264668	0.00356245
200	0.00284958	0.00312233	0.00460935	0.00650167
400	0.01128125	0.01279855	0.01497626	0.02946782
700	0.03598309	0.04372573	0.03701901	0.11635494
1000	0.07523966	0.08385062	0.07248425	0.21952653
2000	0.31524277	0.35352468	0.19600582	0.91470623
3000	0.77258992	0.79924870	0.31492233	2.10295439
4000	1.28199744	1.39382863	0.45474410	3.78487015
5000	2.12303495	2.21864676	0.54675722	6.08174467
7000	4.21341491	4.76713562	0.91526246	11.97331214
8500	5.78077078	6.04004073	1.09886718	18.72166705
10000	7.99377108	8.60798383	1.09604669	25.58029509
15000	18.78544307	20.55864453	2.24437189	56.95080543
20000	37.25268340	41.01605058	2.88679028	107.43589687
25000	54.49094725	60.03861308	3.62000000	154.19083738
30000	73.88243604	81.57779932	4.32849813	220.25348997
40000	141.53288269	147.29634213	6.46274900	443.17752767

Табл. 1.1: Время выполнения сортировок

В таблице **??** приведены результаты выполнения реализаций алгоритмов сортироки на языке Python в зависимости от размера сортируемого массива, где

N — количество элементов в массиве (его размер);

select (1) — сортировка выбором с использованием конструкции *while*;

select (2) — сортировка выбором с использованием конструкции *for... in range()*;

quick — быстрая сортировка;

bubble — сортировка пузырьком.

Из данных в этой таблице построены графики на рисунке **??**, из которых очевидно, что время выполнения сортировки пузырьком растет существенно быстрее остальных сортировок.

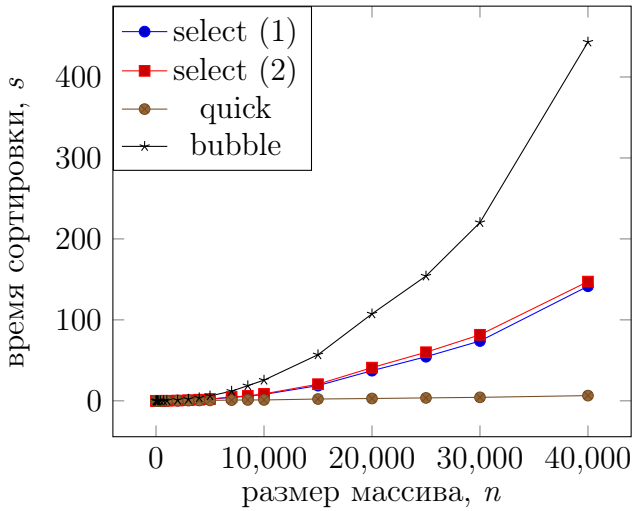

Рис. 1.1: Зависимость времени выполнения алгоритмов от размера массива

При этом можно сделать вывод, что реализация алгоритма с участием конструкции *while* быстрее, чем аналогичная с использованием конструкции *for... in range()*. Последняя, при своей работе, создает множество, перебирая все его элементы, таким образом являясь аналогом *foreach* из некоторых других языков, например PHP.

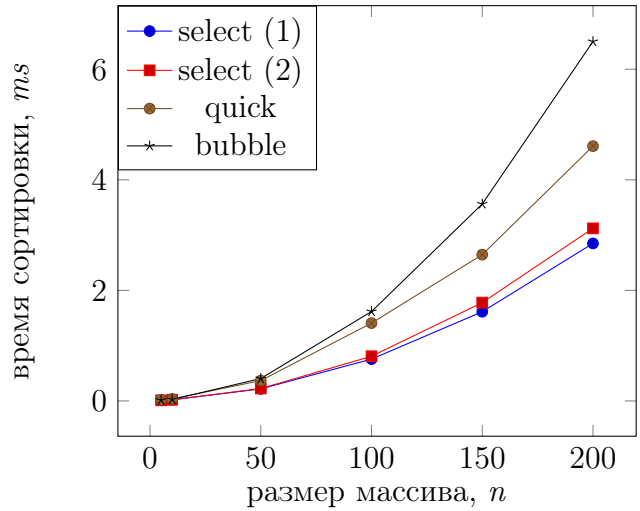

Рис. 1.2: Зависимость времени выполнения от размера для небольших массивов

Из представленных графиков можно сделать вывод, что сортировку пузырьком целесообразно использовать лишь для небольших массивов (порядка десятков элементов).

В то же время видно, что представленная реализация алгоритма быстрой сортировки проигрывает во времени алгоритму сортировки выбором на небольших массивах. Ведь, несмотря на меньшую алгоритмичискую сложность ($O(n \log n)$ против $O(n^2)$), при её оценке не учитывается константа c, которая играет роль при небольших n.

Сенчишен Д.А.