**Построение и анализ алгоритмов**

Отчет по КДЗ 1 «Сортировки»

*by Алексей Родионов, БПИ208*

Типы сортировок

На небольших объемах данных заметна квадратичная сложность сортировки случайного массива. Заметный скачок времени при 270 элементах. На больших объемах данных также видна асимптотика порядка N^2.

Высокие временные затраты наблюдаются при обратно отсортированном массиве небольшого объема из-за большого количества перестановок. На больших объемах данных асимптотика стремится к N^2, но снижается при почти отсортированном массиве из-за малого количества перестановок.

При обратно отсортированном массиве данных наблюдается наихудший результат, так как лишних перестановок все равно много. Имеет схожие результаты с обычной сортировкой пузырьком (наблюдается скачок времени при N=250, на больших объемах сложность квадратичная)

Имеет схожую сложность работы с предыдущими версиями сортировки пузырьком. Наблюдается характерная квадратичная сложность и наихудший результат при обратно отсортированном массиве.

Также наблюдается наихудшее время работы при обратно отсортированном массиве из-за большого количества сравнений и перемещений и средняя сложность в N^2

Наблюдаются схожие результаты с сортировкой обычными вставками. Наихудший результат также при обратно отсортированном массиве. На небольших объемах данных сложность может снизиться до логарифмической, но на больших объемах по-прежнему порядка N^2

Время работы значительно ниже, чем у предыдущих сортировок. На небольших объемах данных сортировка работает за время порядка O(n+k), где k – размер подсчета, что в среднем сводится к O(n). На больших объемах данных время работы скачет по разным значениям, так как выбирается разный максимальный/минимальный элемент массива, однако она гораздо лучше, чем квадратичная.

Время работы при случайных числах от 0 до 5 практически линейное, так как асимптотика работы O(n\*k), где k – разрядность максимального числа. Время работы зачастую колеблется, что связано с различным выбором числа k в зависимости от разрядности максимума., однако почти все случаи кроме 0–5 работают приблизительно за одно и то же время.

Сортировка имеет асимптотику O(n\*logn), что заметно при сравнении с предыдущими графиками, где была практически линейная и квадратичная сложности. На небольших объемах данных при обратно отсортированном массиве наблюдаются скачки, а на больших объемах во всех 4 случаях сортировка показывает стабильное время работы за небольшим исключением колебаний.

Наилучшее время работы у сортировки при обратно отсортированном массиве, однако почти отсортированный массив значительно увеличивает асимптотику. На больших объемах данных и больших случайных числах видна периодическая деградация сортировки, когда при работе асимптотика с O(n\*logn) снижается до O(n^2).

На небольших объемах данных видны периодические ухудшения при случайных больших числах и обратно отсортированном массиве. Из-за выбора последнего элемента в качестве опорного, на больших объемах данных наблюдается деградация асимптотики с логарифмической до квадратичной при обратно отсортированном массиве. Это связано с большим количеством вызовов из-за разделения массива на подмассивы длиной в 1 и n-1 элементов.

Несмотря на небольшие колебания, сортировка стабильно работает при всех возможных вариантах генерации за время порядка n\*logn. В некоторых случаях время аналогично быстрой сортировке, однако сортировка кучей не деградирует до квадратичной. Колебания можно объяснить разным временем при поиске максимума и балансировке дерева в зависимости от заданной последовательности.

Типы массивов

Наихудшее время работы у сортировок пузырьком, имеющий асимптотику O(n^2), видны большие скачки по времени работы. На больших же объемах данных лучше всего из квадратичных себя показывают сортировки выбором, вставками и бинарными вставками, чья асимптотика становится иже, чем у квадратичных пузырьков.

На больших случайных значениях видна похожая ситуация, что и у малых – худший случай – пузырьки, лучший случай – вставки и выбор.

Среднее время работы сортировок снизилось по сравнению с предыдущими случайными массивами, однако общая эффективность квадратичных сортировок осталось почти прежней. Наихудшее время работы по-прежнему у пузырьковых сортировок, а лучшее у вставок, однако асимптотика сортировки выбором приблизилась к квадратичной и стала схожей с пузырьковой.

Ситуация схожа со случайными массивами – положение сортировок почти такое же, однако общее время работы увеличилось. На небольших объемах данных видны недостатки пузырьковых сортировок, которые совершают слишком много лишних перемещений. Лучшим образом себя вновь показывает сортировка выбором, после нее идут вставочные сортировки.

Во всех случаях был смысл сравнивать только квадратичные сортировки, так как эффективные сортировки, работающие за n\*logn показывают слишком хороший результат и их невозможно сравнить.

Выводы

* Наихудшим образом себя показали пузырьковые сортировки, требующие O(1) памяти, однако работающие за время O(n^2).
* Сортировки вставками также требуют O(1) памяти и их время иногда снижается до O(n\*logn), но в большинстве случаев они также неэффективны. Аналогичная ситуация и с сортировкой выбором, которая довольно эффективна, но лишь на некоторых входных данных.
* Таким образом, простые в реализации сортировки, как правило, неэффективны.
* Сортировка подсчетом и цифровая сортировка устойчивы, однако требуют довольно много памяти O(n+k), где k зависит от максимального числа в массиве. Кроме того, цифровая сортировка подходит только для чисел, и иные структуры данных упорядочить ей будет нельзя.
* Сортировка слиянием устойчива и работает за логарифмическое время O(n\*logn), однако требует дополнительно O(n) памяти и зачастую плохо работает на обратно отсортированном массиве.
* Быстрые сортировки эффективны и требуют лишь O(logn) памяти и показывают время работы порядка O(n\*logn), однако неустойчивы, могут вызвать ошибку переполнения стека, если написаны рекурсивно и на неудачных массивах (уже отсортированных/обратно отсортированных/почти отсортированных, в зависимости от выбора опорного элемента) деградируют до O(n^2).
* Сортировка кучей имеет стабильную асимптотику O(n\*logn) и требует константу памяти, но неустойчива.
* В зависимости от конкретной задачи и того, что в ее контексте важнее (время, память или устойчивость), нужно выбирать подходящую сортировку, однако универсальный вариант – использовать быструю сортировку, но при деградации переключаться на сортировку кучей. Именно этот вариант лежит в основе std::sort() языка C++.