Санкт-Петербургский государственный университет

Доклад по предмету “Алгоритмы и анализ сложности”

LSD-поразрядная сортировка

Выполнил бакалавр 3 курса:
Полоз Алексей Евгеньевич

Санкт-Петербург
2018

Содержание

Описание, история разработки, область применения

Поразрядная сортировка Radix Sort не использует сравнений сортируемых элементов.

Есть два вида такой сортировки — MSD (most significant digit — сначала старший разряд) и LSD (least significant digit — сначала младший разряд). LSD несколько удобнее для сортировки чисел, т.к. не приходится «приписывать» к числам слева незначащие 0 для выравнивания числа разрядов. MSD же удобнее для сортировки строк.

Математический анализ

Сравнение производится поразрядно: сначала сравниваются значения одного крайнего разряда. Сортировка в таком случае выполняется с помощью сортировки подсчетом (count sort). Сложность — o(n). Далее элементы группируются по результатам этого сравнения, затем сравниваются значения следующего разряда, соседнего, и элементы либо упорядочиваются по результатам сравнения значений этого разряда внутри образованных на предыдущем проходе групп, либо переупорядочиваются в целом, но сохраняя относительный порядок, достигнутый при предыдущей сортировке. Затем аналогично делается для следующего разряда, и так до конца.

Асимптотическая оценка всего алгоритма: до сортировки необходимо знать два параметра: количество разрядов в самом длинном ключе и разрядность данных — количество возможных значений разряда ключа.

База: n=1. Очевидно, что алгоритм работает верно, потому что в таком случае мы просто сортируем младшие разряды какой-то заранее выбранной устойчивой сортировкой.

Переход: Пусть для n=k алгоритм правильно отсортировал последовательности по k младшим разрядам. Покажем, что в таком случае, при сортировке по (k+1)-му разряду, последовательности также будут отсортированы в правильном порядке. Вспомогательная сортировка разобьет все объекты на группы, в которых (k+1)-й разряд объектов одинаковый. Рассмотрим такие группы. Для сортировки по отдельным разрядам мы используем устойчивую сортировку, следовательно порядок объектов с одинаковым (k+1)-м разрядом не изменился. Но по предположению индукции по предыдущим k разрядам последовательности были отсортированы правильно, и поэтому в каждой такой группе они будут отсортированы верно. Также верно, что сами группы находятся в правильном относительно друг друга порядке, а, следовательно, и все объекты отсортированы правильно по (k+1)-м младшим разрядам.

Шагов столько, сколько разрядов в числах. Пусть m — количество разрядов, n — количество объектов, которые нужно отсортировать, T(n) — время работы устойчивой сортировки. Цифровая сортировка выполняет k итераций, на каждой из которой выполняется устойчивая сортировка и не более O(1) других операций. Следовательно время работы цифровой сортировки — O(kT(n)). Рассмотрим отдельно случай сортировки чисел. Пусть в качестве аргумента сортировке передается массив, в котором содержатся n m-значных чисел, и каждая цифра может принимать значения от 0 до k−1. Тогда цифровая сортировка позволяет отсортировать данный массив за время O(m(n+k)), если устойчивая сортировка имеет время работы O(n+k). Если k небольшое, то оптимально выбирать в качестве устойчивой сортировки сортировку подсчетом. Если количество разрядов — константа, а k=O(n), то сложность цифровой сортировки составляет O(n), то есть она линейно зависит от количества сортируемых чисел.

Алгоритму в такой реализации требуется дополнительная память — O(n).

Характеристика данных

Алгоритм изначально был предназначен для сортировки целых чисел, записанных цифрами. Но так как в памяти компьютеров любая информация записывается целыми числами, алгоритм пригоден для сортировки любых объектов, запись которых можно поделить на «разряды», содержащие сравнимые значения.

Примерами объектов, которые удобно разбивать на разряды и сортировать по ним, являются числа и строки.

- Для чисел уже существует понятие разряда, поэтому будем представлять числа как последовательности разрядов. Конечно, в разных системах счисления разряды одного и того же числа отличаются, поэтому перед сортировкой представим числа в удобной для нас системе счисления.
- Строки представляют из себя последовательности символов, поэтому в качестве разрядов в данном случае выступают отдельные символы, сравнение которых обычно происходит по соответствующим им кодам из таблицы кодировок. Для такого разбиения самый младший разряд — последний символ строки.

Реализация программы

Пример реализации в личном репозитории:
https://github.com/kosyachniy/spbu/blob/master/algorithm_complexity/lsd.py

Также программа, генерирующая данные и вызывающая сортировку: https://github.com/kosyachniy/spbu/blob/master/algorithm_complexity/main.py
В этом файле также есть функция graph для построения графиков в зависимости от разных входных данных.

Входные данные

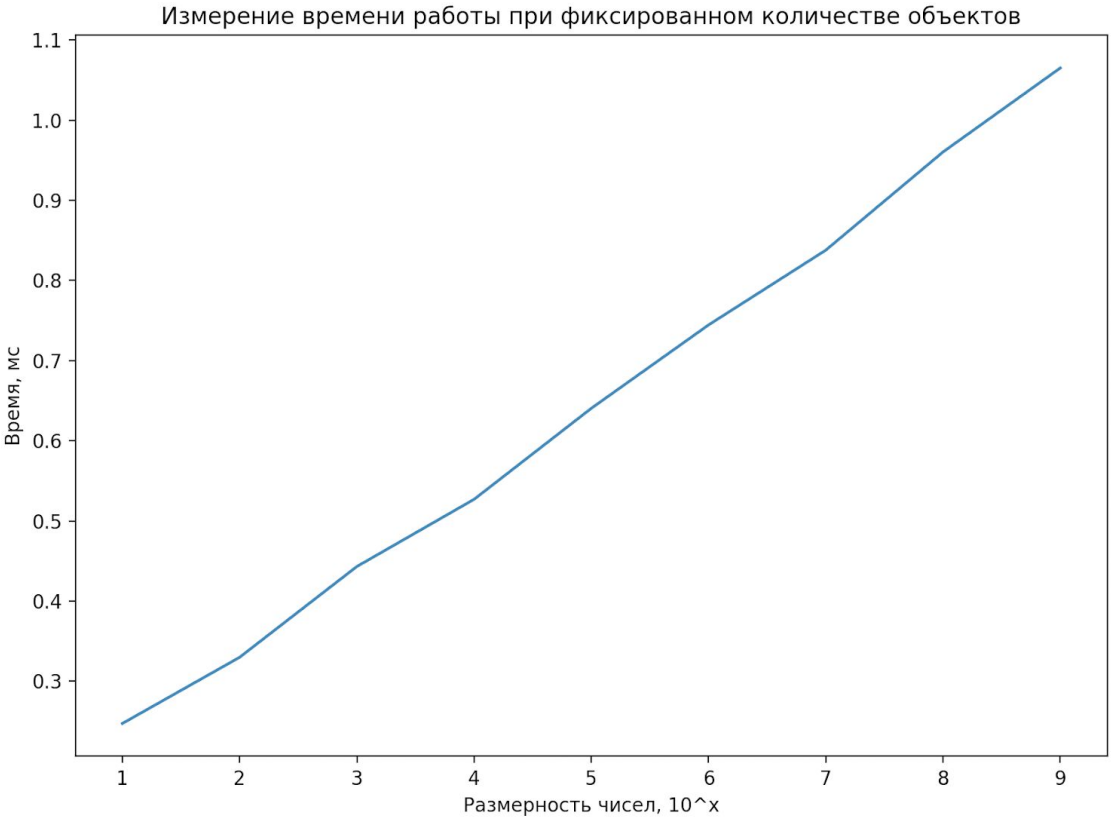
Входные данные в конкретной реализации представляют из себя массив десятичных чисел.

Для чистоты эксперимента можно брать числа разной длины, повторяющейся длины, повторяющиеся числа, с повторяющимися цифрами в разрядах. В следующем пункте приведены примеры измерений с разными входными данными.

Генерация данных — функция gen_digit в файле: https://github.com/kosyachniy/spbu/blob/master/algorithm_complexity/main.py

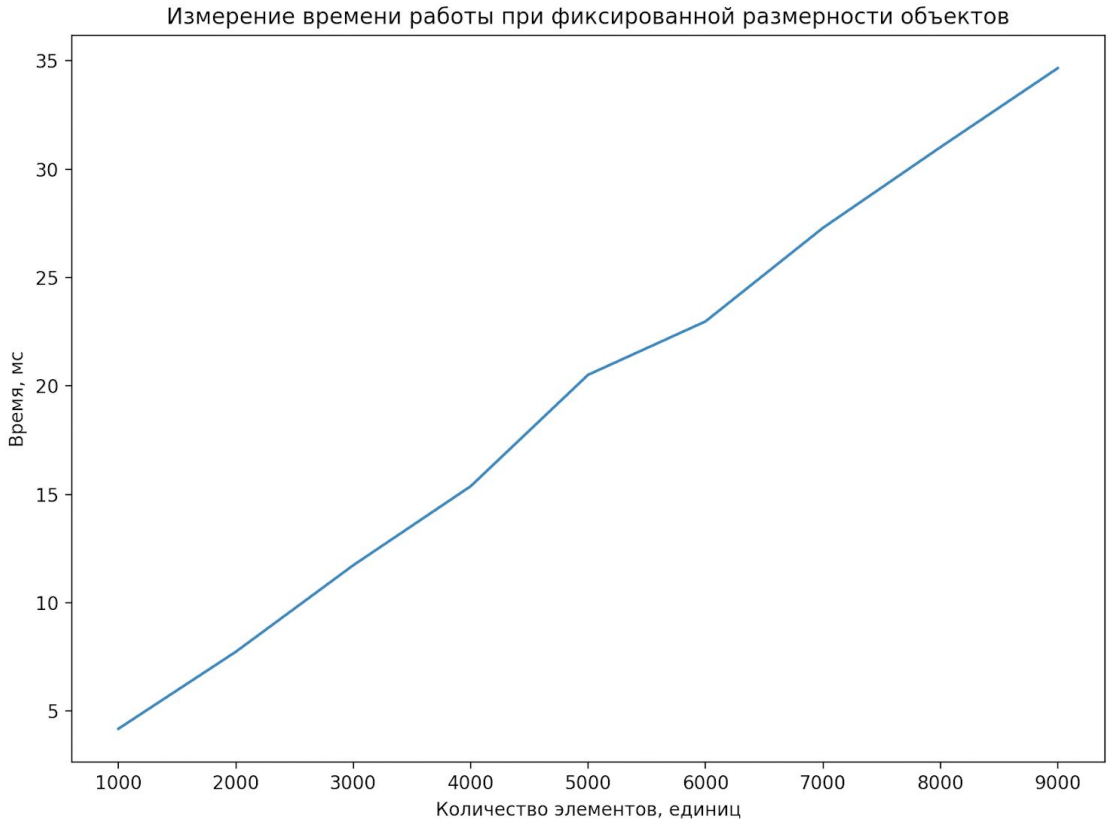
Измерение

Количество разрядов m=10
Зависимость от n:



Данный график отображает линейную зависимость с погрешностями вычисления.

Количество элементов n=1000
Зависимость от m:



Данный график отображает линейную зависимость с погрешностями вычисления.

Используемые источники

- https://habr.com/post/268261/
- http://algolist.manual.ru/sort/radix_sort.php
- https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D0%B0%D0%B7%D1%80%D1%8F%D0%B4%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0#%D0%A0%D0%B5%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_(LSD)
- https://neerc.ifmo.ru/wiki/index.php?title=%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0
- http://rosettacode.org/wiki/Sorting_algorithms/Radix_sort
- https://habr.com/post/335920/