Лабораторная работа № 1 по курсу дискретного анализа: сортировка за линейное время

Выполнила студентка группы 08-208 МАИ *Шевлякова София*.

Условие

1. Требуется разработать программу, осуществляющую ввод пар «ключ-значение», их упорядочивание по возрастанию ключа указанным алгоритмом сортировки за линейное время и вывод отсортированной последовательности.

2. Вариант задания: N. 5-2
   Тип сортировки: поразрядная сортировка
   Тип ключа: MD5-суммы (32-разрядные шестнадцатиричные числа)
   Тип значения: строки переменной длины (до 2048 символов)

Метод решения

Поразрядная сортировка имеет 2 версии, в этой лабораторной я решила использовать LSD (Least Significant Digit radix sort). При этом, сортировка элементов одного разряда может происходить любым образом, главное - чтобы эта сортировка была устойчивой, я выбрала сортировку подсчетом. Так как ключ представляет собой шестнадцатеричное число, классическую сортировку подсчетом мне пришлось доработать для обработки символов.

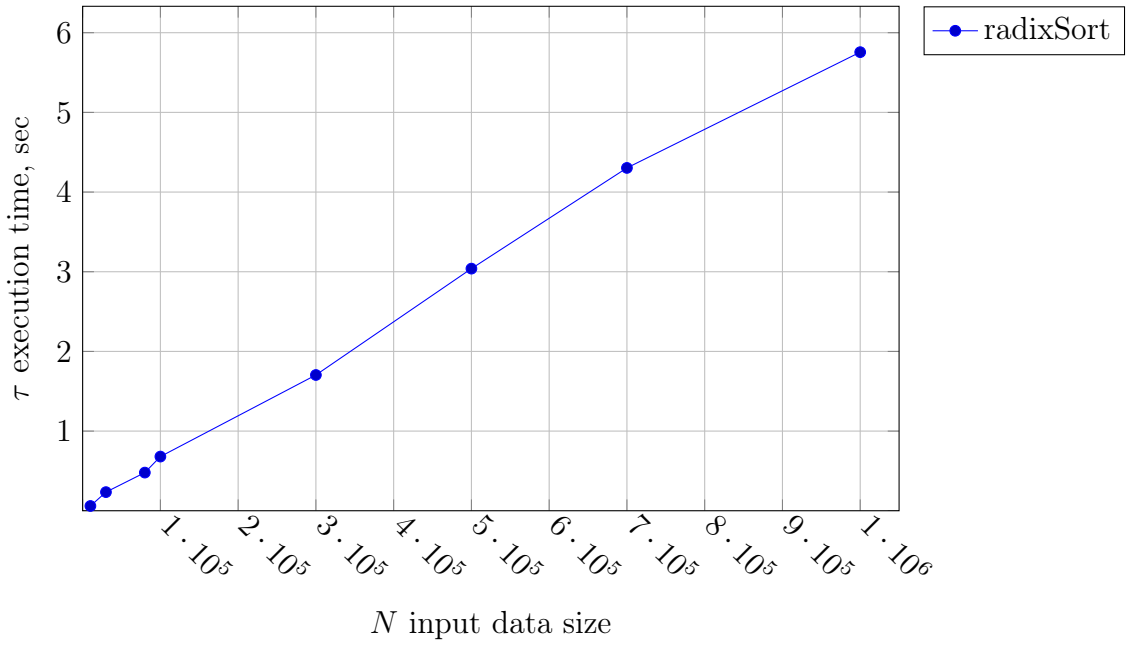
Описание программы

Для удобного хранения элементов вида ключ-значение составлена структура данных pair, хранящая строку ключа char key[MAX_KEY] и структуру с индексами начала и конца строки значения в непрерывном массиве, который хранит все значения элементов allData. Благодаря такому подходу, мы можем не создавать для каждого элемента массив символов на 2048 элементов, а значит оптимально используем память.
Основная функция программы void radix_sort(struct pair *inputed, int index) сортирует массив inputed при помощи поразрядной сортировки, index - количество входных элементов, то есть размер массива inputed. По сути, мы несколько раз вызываем функцию сортировки подсчётом void counting_sort(struct pair *inputed, int i, int index), которая сортирует массив inputed, по разряду i. Внутри эта функция создает новый массив res, в него она записывает отсортированный массив, а потом копирует его в исходный массив inputed.

Дневник отладки

№	Описание ошибки	Способ устранения
1	Так как я создавала статический массив структур, одним из элементов которой является массив на 2048 символов, то максимальный размер массива, который я могла задать, был 4000, что оказалось слишком мало.	Для исправления этой ошибки я попробовала выделять память при помощи malloc, так как в этом случае память выделяется из кучи, я смогла увеличить массив до 9000.
2	Но массива на 9000 оказалось все равно мало для прохождения тестов. Писать свой вектор мне не очень хотелось, поэтому я продолжила искать способ увеличения размера массива.	Для исправления я решила создать массив символов, в котором буду хранить строку, состоящую из значений всех пар, а в структуре буду теперь хранить индекс начала и конца значения для этого ключа.
3	Программа наконец-то прошла 11 тест, но в последующих тестах входные данные только увеличивались, что снова вызывало ошибку.	Для исправления я просто меняла значение констант, отвечающие за размер массива, который состоит из значений каждой пары, и размер общего массива структур.

Тест производительности



Оценка сложности алгоритма: O(n · L), где L - число блоков (размер элемента), n - количество элементов в массиве. В нашем случае, L = 32, при постоянном L алгоритм будет линеен относительно количества входных данных.
Как видно из графика, рост времени работы при увеличении объема входных данных в среднем увеличивается линейно.

Выводы

Выполняя лабораторную работу по курсу «Дискретный анализ», я впервые работала с языком C++, научилась реализовывать поразрядную сортировку и сортировку подсчётом, вспомнила работу с памятью. Это поможет мне в ситуации, когда нужно будет написать быструю сортировку, которая будет работать за линейное время. Также я узнала, что *malloc* выделяет память из кучи, а память под статический массив выделяется из стека.