Московский авиационный институт (национальный исследовательский университет)

Факультет компьютерных наук и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №1 по курсу «Дискретный анализ»

Студент: Л. А. Постнов
Преподаватель: А. А. Кухтичев
Группа: М8О-206Б-22

Дата:
Оценка:
Подпись:

Москва, 2024

Лабораторная работа №1

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

Вариант сортировки: Сортировка подсчетом.

Вариант ключа: `числа от 0 до 65535.`

Вариант значения: `строки переменной длины (до 2048 символов).`

1 Описание

Основная идея сортировки подсчетом заключается в том, чтобы для каждого входного элемента x определить количество элементов, которые меньше x. С п омощью этой информации элемент x можно разместить на той позиции выходного массива, где он должен находиться [1].

При составлении кода для этого алгоритма предполагается, что на вход подается массив $A[1..n]$, так что $length[A] = n$. Потребуются еще два массива: в массиве $B[1..n]$ будет содержаться отсортированная выходная последовательность, а массив $C[0..k]$ служит временным рабочим хранилищем [1].

Сложность данной сотрировки: $\Theta(n + k)$.

2 Исходный код

В качестве входных данных имеется пара «ключ-значение», поэтому разумно создать структуру `Pair`, полями которой будет ключ типа `unsigned short` и значение типа `std::string`.

```
struct Pair{
    unsigned first;
    std::string second;
};
```

Сортировка подсчетом имеет слудующую реализацию, согласно алгоритму, описанному в [1]. На вход по ссылке подается исходная последовательность пар «ключ-значение», хранимая в кастомном классе `vector::Vecotor`.

```
namespace sort{
    void counting_sort(vector::Vector<Pair> &elems){

        if(elems.empty()){
            return;
        }

        unsigned short max_elem = 0;
        for(size_t i = 0; i < elems.size(); ++i){
            max_elem = (elems[i].first > max_elem) ? elems[i].first : max_elem;
        }

        int tmp[max_elem + 1] = {0};

        for(size_t i = 0; i < elems.size(); ++i){
            ++tmp[elems[i].first];
        }

        for (size_t i = 1; i < max_elem + 1; ++i){
            tmp[i] += tmp[i - 1];
        }

        vector::Vector<Pair> res(elems.size());
        for(int i = elems.size() - 1; i >= 0; --i){
            size_t pos = tmp[elems[i].first]--;
            res[pos-1] = elems[i];
        }
        elems = res;
    }
}
```

3 Консоль

```
wednees@MSI:/mnt/c/Users/leoni/OneDrive/Desktop/study/Discrete_Analysis/
lab1/solutions$ g++ solution.cpp -o solution
wednees@MSI:/mnt/c/Users/leoni/OneDrive/Desktop/study/Discrete_Analysis/
lab1/solutions$ ./solution
1       a
1       b
0       c
4       g

0       c
1       a
1       b
4       g
wednees@MSI:/mnt/c/Users/leoni/OneDrive/Desktop/study/Discrete_Analysis/
lab1/solutions$ ./solution
0       xGfxrxGGxrxMMMMfrrrG
65535   xGfxrxGGxrxMMMMfrrr
0       xGfxrxGGxrxMMMMfrr
65535   xGfxrxGGxrxMMMMfr

0       xGfxrxGGxrxMMMMfrrrG
0       xGfxrxGGxrxMMMMfrr
65535   xGfxrxGGxrxMMMMfrrr
65535   xGfxrxGGxrxMMMMfr
```

4 Тест производительности

Тесты производительности представляют из себя следующее: сортировка подсчетом сравнивается с *stable sort* из стандартной библиотеки шаблонов на тестах из 10^4, 10^5 и 10^6 элементов.

```
Count of lines is 10000
Counting sort time: 1103us
STL stable sort time: 5735us

Count of lines is 100000
Counting sort time: 10180us
STL stable sort time: 81780us

Count of lines is 1000000
Counting sort time: 137663us
STL stable sort time: 1062711us
```

Как видно, на всех тестах скорость работы сортировки подсчетом намного быстрее сортировки из стандартной библиотеки шаблонов, причес с количесвом тестовых значений растет и разница между сортировками. Получается так, потому что сортировка подчетом имеет сложность $\Theta(n + r)$ (где n - количесвто элементов, а r - максимальный элемент), в свою очередь *stable sort* работает за $\Theta(n * log(n))$.

5 Выводы

Выполнив первую лабораторную работу по курсу «Дискретный анализ», я узнал о некоторых алгоритмах устойчивых сортировок за линейное время, подробнее разобрался и самостоятельно реализовал алгоритм сортировки поддсчетом. Выполняя эту задачу, я получил практические навыки в написании структур, а менно собственной реализации шаблонного вектора на языке C++. Отдельно нужно отметить важность в тестировании программы, без использования тестирующей системы, в моем случае, это помогл выявить и исправить ошибку, возникающую при большом количестве входных данных.

Список литературы

[1] Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. *Алгоритмы: построение и анализ, 2-е издание.* — Издательский дом «Вильямс», 2007. Перевод с английского: И. В. Красиков, Н. А. Орехова, В. Н. Романов. — 1296 с. (ISBN 5-8459-0857-4 (рус.))