Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №1 по курсу «Дискретный анализ»

Студент: Р. Р. Гаптулхаков
Преподаватель: Н. С. Капралов
Группа: М8О-208Б

Дата:
Оценка:
Подпись:

Москва, 2020

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: Поразрядная сортировка.

Вариант ключа: `Телефонные номера, с кодами стран и городов в формате +<код страны> <код города> телефон.`

Вариант значения: `Числа от 0 до` $2^{64} - 1$.

1 Описание

В информатике, радикс рода является не-сравнительного алгоритмом сортировки. Он избегает сравнения, создавая и распределяя элементы по сегментам в соответствии с их основанием. Для элементов с более чем одной значащей цифрой этот процесс сегментирования повторяется для каждой цифры с сохранением порядка предыдущего шага, пока не будут учтены все цифры. По этой причине радикальную сортировку также называют сортировкой по корзине и цифровой сортировкой [1]

2 Исходный код

Здесь должно быть подробное описание программы и основные этапы написания кода.

На каждой непустой строке входного файла располагается пара «ключ-значение», поэтому создадим новую структуру TPair, в которой будем хранить ключ и значение. В моём варианте, структура состоит из двух полей: char[] key, unsigned long long value. Затем нужно реализовать динамичесий массив (class TVector), который будет хранить множество пар «ключ-значение». В функции main, с помощью метода Push_back() класса TVector, мы добавляем в наш динамический массив элементы структуры TPair. Затем, с помощью функции Counting_sort(), мы сортируем все элеметны нашего класса TVector. Функция DeleteZero() удаляет все незначащие нули нашего номера перед выводом результата на экран.

main.c	
void Counting_sort(TVector<TPair>& MyData)	Функция поразрядной сортировки
file1.c	
void DeleteZero(TVector<TPair>& MyData)	Функция, которая удаляет лишние нули в номере.

```
1
2  struct TPair{
3      TPair(){}
4      //  .
5      TPair(char n[], unsigned long long &v);
6      ~TPair(){}
7      unsigned long long value;
8      char number[20];
9      short size;
10 };
11 class TVector{
12 public:
13     TVector();
14     TVector(unsigned int size);
15     TVector(unsigned int size, const T & initial);
16     int Capacity() const;
17     int Size() const;
18     bool Empty() const;
19     T* Begin();
20     T* End();
21     T& Front() const;
22     T& Back() const;
23     void Reserve(unsigned int size);
24     void Push_back(T const &value);
25     void Pop_back();
```

```
26      void Resize(unsigned int size);
27      T& operator[](unsigned int index);
28      TVector<T>& operator=(const TVector<T>& v);
29      ~TVector();
30      void clear();
31  private:
32      unsigned int my_size;
33      unsigned int my_capacity;
34      T* buffer;
35      };
```

3 Консоль

```
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/continuous/DA/solution$ make
g++ -std=c++14 -O2 -Wextra -Wall -Werror -Wno-sign-compare -Wno-unused-result
-pedantic main.cpp -o solution
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/continuous/DA/solution$ cat
test.txt
+7-495-1123212  13207862122685464576
+375-123-1234567        7670388314707853312
+7-495-1123212  4588010303972900864
+375-123-1234567        12992997081104908288
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/continuous/DA/solution$ ./solution
<test.txt
+7-495-1123212  13207862122685464576
+7-495-1123212  4588010303972900864
+375-123-1234567        7670388314707853312
+375-123-1234567        12992997081104908288
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/continuous/DA/solution$ make
clean
rm -rf solution
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/continuous/DA/solution$
```

4 Тест производительности

Тест производительности представляет из себя следующее: считываются данные, записываются в два разных вектора и выполняется сортировка двумя алгоритмами. Вначале выполняется поразрядная сортировка, потом алгоритмом std::stable_sort(), при этом измеряется время работы каждой сортировки по отдельности. На экран выводится количство обработанных строк "ключ-значение" время работы поразрядной сортировки, время работы std::stable_sort().

```
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$ python3 generator.py
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$ python3 generator.py
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$ python3 generator.py
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$ g++ benchmark.cpp
-o ben
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$ ./ben <test1.txt
Count of lines is 10000
Counting sort time: 7037ms
STL stable sort time: 5044ms
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$ ./ben <test2.txt
Count of lines is 1000000
Counting sort time: 1343216ms
STL stable sort time: 817709ms
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$ ./ben <test3.txt
Count of lines is 10000000
Counting sort time: 13694274ms
STL stable sort time: 9759699ms
rusya@DESKTOP-K0H2IC0:/mnt/c/Users/fynex/Desktop/Benchmark$
```

Как видно, что std::stable_sort() выиграл у поразрядной сортировки, так как моя реализация сортировки не до конца оптимизированна.

5 Выводы

Выполнив первую лабораторную работу по курсу «Дискретный анализ», я научился многому. Во-певрвых, я немного познакомился с шаблонами, классами, констуркторами, дестуркторами, узнал, что такое полиморфизм и инкапсуляция. Во-вторых, научился реализовывать конструкторы копирования, перегрузку операторов. В-третьих, все ранее перечисленное, я использовал в своей лабораторной работе, что помогло мне на практике закрепить свои знания. Также научился реализовывать поразрядную сортировку.

Список литературы

[1] Поразрядная сортировка (дата обращения: 01.10.2020).

[2] Реализация вектора. (дата обращения: 01.10.2020).

[3] Конструкторы и деструкторы. (дата обращения: 02.10.2020).

[4] Шаблоны в C++ (дата обращения: 02.10.2020).

[5] Список использованных источников оформлять нужно по ГОСТ Р 7.05-2008