Moscow авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №1 по курсу «Дискретный анализ»

Студент: М. М. Касимов
Преподаватель: А. А. Кухтичев
Группа: М8О-206Б

Дата:
Оценка:
Подпись:

Москва, 2019

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 подсчётом.

Вариант ключа: Числа от 0 до 65535.

Вариант значения: Строки фиксированной длины 64 символа, во входных данных могут встретиться строки меньшей длины, при этом строка дополняется до 64-х нулевыми символами, которые не выводятся на экран.

1 Описание

Основная идея сортировки подсчётом заключается в том, чтобы для каждого входного элемента x определить количество элементов, которые меньше x. С помощью этой информации элемент x можно разместить в той позиции выходного массива, где он должен находиться [1]. Эта сортировка применяется для целых чисел, она имеет линейную сложность $O(n+k)$, где n - количество входных элементов, k - максимальный элемент во входных данных. У данной сортировки есть несколько вариантов реализации. В данной лабораторной работе написана её устойчивая вариация.

2 Исходный код

main.cpp	
TVector<KV> Counting(const TVector<KV> &v, int max)	Функция сортировки подсчётом
void Filling(char *c)	Функция, которая заполняет массив из 65 элементов типа char символами ”.
int main()	Главная функция, в которой происходит чтение данных, вызов функции сортировки и вывод.
vector.h	
TVector(const TVector &o), TVector(size_t size), TVector()	Конструкторы класса TVector.
∼ Tvector()	Деструктор класса TVector.
void Push_back(const T &elem)	Метод класса TVector, который добавляет новый элемент в вектор, а в случае необходимости динамически выделяет дополнительную память.
TVector &operator=(const TVector &o)	Перегрузка оператора присвоения.
size_t Size() const	Метод класса TVector, который возвращает длину нашего вектора.
std::istream &operator»(std::istream &is, KV &elem)	Перегрузка оператора ввода.
std::ostream &operator«(std::ostream &is, KV &elem)	Перегрузка оператора вывода.
T &operator[ ](size_t number)	Перегрузка оператора квадратные скобки, для прямого доступа к элементам вектора.

На каждой непустой строке входного файла располагается пара «ключ-значение», поэтому создадим новую структуру KV, в которой будем хранить ключ и значение.

```
1 struct KV {
2 unsigned short key;
3 char value[65];
4 };
```

Мы не знаем количество входных данных, поэтому мы напишем динамический массив - вектор, в который будут помещаться структуры KV.

```
template<typename T>
class TVector {
public:
  TVector() = default;
  TVector(const TVector &o);
  TVector(size_t size);
  ~TVector();
  T &operator[](size_t number);
  const T &operator[](size_t number) const;
  void Push_back(const T &elem);
  TVector &operator=(const TVector &o);
  size_t Size() const;
private:
    T *Data = nullptr;
    size_t SizeVector = 0;
    size_t Capacity = 0;
};
```

3 Консоль

```
magomed@magomed-pc ~/da_lab1/build $ vim test1
magomed@magomed-pc ~/da_lab1/build $ vim test2
magomed@magomed-pc ~/da_lab1/build $ make
[100%] Built target da_lab1
magomed@magomed-pc ~/da_lab1/build $ cat test1
4 a
3 g
2 a
1 m
magomed@magomed-pc ~/da_lab1/build $ ./da_lab1 <test1
1 m
2 a
3 g
4 a
magomed@magomed-pc ~/da_lab1/build $ cat test2
0 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naatt
65535 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naat
0 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naa
65535 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3na
magomed@magomed-pc ~/da_lab1/build $ ./da_lab1 <test2
0 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naatt
0 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naa
65535 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naat
65535 n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3na
magomed@magomed-pc ~/da_lab1/build $ cat test3
magomed@magomed-pc ~/da_lab1/build $ ./da_lab1 <test3
```

4 Тест производительности

Тест производительности представляет из себя следующее: мы будем сравнивать время выполнения программы с сортировкой подсчётом с временем выполнения программы с библиотечной сортировкой sort(), она же быстрая сортировка. Для этого я сгенерировал два теста, в первом 100000 строк, а во втором 1000000. А поможет нам в этом утилита time, которая покажет время выполнения программы.

```
magomed@magomed-pc ~/da_lab1/build $ wc -l 03.t
100000 03.t
magomed@magomed-pc ~/da_lab1/build $ wc -l 05.t
1000000 05.t
magomed@magomed-pc ~/da_lab1/build $ time ./da_lab1 <03.t

real 0m0.088s
user 0m0.076s
sys 0m0.004s
magomed@magomed-pc ~/da_lab1/build $ time ./a.out <03.t

real 0m0.134s
user 0m0.112s
sys 0m0.012s
magomed@magomed-pc ~/da_lab1/build $ time ./da_lab1 <05.t

real 0m0.647s
user 0m0.616s
sys 0m0.020s
magomed@magomed-pc ~/da_lab1/build $ time ./a.out <05.t

real 0m1.372s
user 0m1.328s
sys 0m0.028s
```

Мы видим, что сортировка подсчётом выиграла по скорости у быстрой сортировки. Неудивительно, ведь у сортировки подсчётом линейная сложность по времени, а у быстрой сортировки $O(nlog(n))$. Справедливости ради, стоит отметить, что сортировка подсчётом работает только для целых чисел и она является неприемлимой в тех случаях, когда максимальный элемент последовательности оказывается намного больше количества элементов последовательности.

5 Выводы

Выполнив первую лабораторную работу по курсу «Дискретный анализ», я научился пользоваться утилитами: time, cmake, valgrind. Также я узнал новые сортировки, которые работают за линейное время. Улучшил навыки отладки своей программы. Освоил довольно приятный набор правил, по которому писать свою программу гораздо легче.

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.))

[2] *Сортировка подсчётом — Википедия.*
URL: `http://ru.wikipedia.org/wiki/Сортировка_подсчётом` (дата обращения: 16.12.2013).