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: Г. А. Ермеков
Преподаватель: С. А. Михайлова
Группа: М8О-201Б
Дата:
Оценка:
Подпись:

Москва, 2025

Лабораторная работа №1

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: Автомобильные номера в формате A 999 BC (используются буквы латинского алфавита).

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 алгоритма решения задачи состоит в применении поразрядной сортировки, которая позволяет упорядочивать автомобильные номера в формате A 999 BC за линейное время благодаря их структурированности. Для реализации необходимо обработать ключи по разрядам в порядке от младшего к старшему, начиная с цифр, затем букв, при этом на каждом этапе применяется сортировка подсчетом для текущего разряда. Сортировка подсчетом идеально подходит, поскольку разряды ключей ограничены фиксированным количеством значений (10 для цифр и 26 для каждой из буквенных групп). Тип значения, представляющий строки фиксированной длины до 64 символов, требует предварительной нормализации путем добавления нулевых символов, чтобы все строки стали одинаковой длины. После нормализации строки участвуют в сортировке как привязанные к ключам данные и выводятся в соответствии с упорядоченными ключами, при этом нулевые символы не отображаются в результате.

2 Исходный код

На каждой непустой строке входного файла располагается пара «ключ-значение», в которой ключ указан согласно заданию, затем следует знак табуляции и указано соответствующее значение.

[language=C] include <bits/stdc++.h> using namespace std;

struct Entry array<uint8$_t$, 6 > key; $charvalue$[65]; ;

int main() ios::sync$_w ith_s tdio(false)$; $cin.tie(nullptr)$;

vector<Entry> entries; string line;

while (getline(cin, line)) if (line.empty()) continue; auto tab = line.find('$\widehat{}$');

string key$_s$; $key_s.reserve(6)$; $for(charc : line.substr(0, tab))if(isalnum(static_cast < unsignea$
$6)continue$;

Entry e;

auto val = line.substr(tab + 1); size$_t len = min(val.size(), size_t(64))$; $memcpy(e.value, val.dat$
$'$;

for (int i = 0; i < 6; ++i) char c = key$_s$[i]; $e.key[i] = isalpha(static_cast <$
$unsignedchar > (c))?c -' A' : c -' 0'; entries.push_back(e)$;

size$_t n = entries.size()$; $vector < Entry > buffer(n); array < int, 26 >$
$cnt; array < int, 26 > pos_arr$;

for (int d = 5; d >= 0; –d) int range = (d == 0 || d >= 4 ? 26 : 10);
cnt.fill(0); for (auto e : entries) cnt[e.key[d]]++; pos$_a$rr[0] = 0; $for(inti =$
$1; i < range; + + i)pos_arr[i] = pos_arr[i - 1] + cnt[i - 1]$;

for (auto e : entries) int k = e.key[d]; buffer[pos$_a$rr[k]++] = e; $entries.swap(buffer)$;

for (auto e : entries) cout « char('A' + e.key[0]) « ' '; cout « int(e.key[1])
« int(e.key[2]) « int(e.key[3]) « ' '; cout « char('A' + e.key[4]) « char('A' +
e.key[5]); cout « '$\widehat{}$' « e.value « "; return 0;

3 Консоль

george@GEORGE-PC:/mnt/c/Users/george/Projects/MaiLabs/MAI$_l abs_D iscran/src$ make main g++ -O2 -std=c++20 -Wall -Wextra -o main main.cpp george@GEORGE-PC:/mnt/c/Users/george/Projects/MaiLabs/MAI$_l abs_D iscran/src$ ./main A 000 AA n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naatt
Z 999 ZZ n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naat
A 000 AA n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naa
Z 999 ZZ n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3na
A 000 AA n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naatt
A 000 AA n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naa
Z 999 ZZ n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3naat
Z 999 ZZ n399tann9nnt3ttnaaan9nann93na9t3a3t9999na3aan9antt3tn93aat3na

4 Тест производительности

Тест производительности заключается в сравнении поиска образца при помощи Z-алгоритма и стандартной функции строки `std::find()` в языке C++. В данном случае не учитывается время на чтение входных данных. Основная строка состоит из 1 миллиона букв, всего примерно 200 образцов длиной от 2 до 100 символов.

[george@GEORGE-HONOR14 src]$g++zblock.cpp-O2$[$george@GEORGE-HONOR14src$] ./a.out < ../input.txt > zout.txt [george@GEORGE-HONOR14 src]$catzout.txt|grep"time"Z-algorithmsearchtotaltime$: 0.025122sec[$george@GEORGE-HONOR14src$] g++ stdfind.cpp -O2 [george@GEORGE-HONOR14 src]$./a.out <$ $../input.txt > findout.txt$[$george@GEORGE-HONOR14src$] cat findout.txt | grep "time"std::find search total time: 1.374654 sec

При поиске всех 200 образцов Z-алгоритм оказался значительно быстрее стандартной функции `std::find`, так как используется эффективная обработка всей строки за линейное время относительно её длины. Полученная скорость объясняется тем, что `std::find` реализует простой перебор без предварительной обработки шаблона, в то время как Z-алгоритм учитывает префиксные совпадения и работает за $O(|text|+|pattern|)$ для каждого поиска.

Таким образом, для массового поиска образцов Z-алгоритм рекомендуется для больших текстов.

5 Выводы

Выполнив первую лабораторную работу по курсу Дискретный анализ, я изучил radix сортировку и научился её применять на больших объемах 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.))

[2] *Z-функция. ИТМО*
URL: `https://neerc.ifmo.ru/wiki/index.php?title=Z-функция` (дата обращения: 17.05.2025).