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: М. О. Чапалда
Преподаватель: С. А. Михайлова
Группа: М8О-201Б-22

Дата: _____
Оценка: _____
Подпись: _____

Москва, 2024

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 определяется типом ключа (и соответствующим ему методом сортировки) и типом значения:

   (a) Поразрядная сортировка.

   (b) Тип ключа: телефонные номера, с кодами стран и городов в формате +<код страны> <код города> телефон.

   (c) Тип 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.

Описание

Алгоритм поразрядной сортировки (Radix Sort) — это несравнительный алгоритм сортировки, который сортирует элементы, последовательно рассматривая каждый разряд числа. Алгоритм работает следующим образом:

1. Разделение чисел по разрядам: Начиная с наименее значимого разряда (например, единиц) и заканчивая наиболее значимым, алгоритм проходит через каждый разряд.

2. Сортировка элементов по текущему разряду: На каждом шаге элементы сортируются в соответствии с их значением в текущем разряде. Это может быть выполнено с использованием стабильной сортировки, такой как сортировка подсчётом (Counting Sort).

Исходный код

Код выполняет сортировку записей с телефонными номерами с использованием поразрядной сортировки (Radix Sort). Вот алгоритм:

1. Инициализация и Ввод Данных: - Программа начинается с выключения синхронизации потоков ввода/вывода C++ с стандартными потоками C для ускорения ввода и вывода.

- Создается вектор records для хранения записей с телефонными номерами.

- В цикле чтения из стандартного ввода (до EOF или ошибки) программа считывает пары строк, представляющие телефонный номер и связанное с ним значение, и сохраняет их в векторе records как объекты структуры PhoneRecord. При создании PhoneRecord, из телефонного номера удаляются все нецифровые символы, создавая только цифровую строку numericPhone.

2. Поразрядная Сортировка (Radix Sort): - Определяется максимальное количество цифр maxDigits, которое возможно в телефонном номере. В данном случае предполагается, что это 15 цифр.

- Создается вектор индексов indices, который первоначально заполняется последовательными числами от 0 до (records.size() - 1), что соответствует индексам элементов в records.

- В цикле от 0 до maxDigits-1 (для каждой разрядной позиции по порядку от младшего к старшему):

- Создается временный вектор векторов (buckets) для 10-ти цифровых "ведер"(0-9).

- Для каждого индекса из indices извлекается соответствующая цифра из numericPhone для текущего разряда. Если текущий разряд выходит за пределы длины numericPhone, используется 0.

- Индексы записей добавляются в соответствующее "ведро".

- После обработки всех записей indices обновляется путем последовательного добавления индексов из каждого "ведра начиная с 0 и далее по возрастанию.

- После завершения цикла поразрядной сортировки создается новый вектор sortedRecords, в который перемещаются отсортированные записи в соответствии с порядком индексов в indices.

- Оригинальный вектор records заменяется на sortedRecords.

3. Вывод отсортированных анных:

- В конце программа выводит телефон и связанное значение для каждой записи в отсортированном векторе records.

Недочёты

Данная программа для радикс сортировки телефонных номеров страдает от недостатков, связанных с неэффективным использованием памяти из-за дополнительных векторов и фиксированного ограничения на количество цифр, что замедляет обработку больших объемов данных. Улучшения могут включать напрямую обработку исходных строк для уменьшения использования памяти, оптимизацию структуры данных и алгоритмов сортировки для повышения производительности, а также гибкое управление разрядностью для адаптации к различным наборам данных, что сделает программу более эффективной и гибкой.

Выводы

Изучая эту программу, я расширила свои знания в области объектно-ориентированного программирования на C++, углубилась в работу со стандартной библиотекой шаблонов (STL) и освоила поразрядную сортировку. Этот опыт значительно улучшил мое понимание алгоритмов и структур данных, а также развил мои навыки в оптимизации и проектировании программ. Благодаря этому я научилась выбирать наиболее подходящие алгоритмы для эффективной обработки данных, что играет ключевую роль в повышении производительности и эффективности использования ресурсов в моих программных проектах.

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] *Поразрядная сортировка — Википедия.*
URL: ://ru.wikipedia.org/wiki/Поразрядная-соритровка