



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ

«Информатика и системы управления»

КАФЕДРА

«Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

По лабораторной работе №1

По курсу: «Анализ алгоритмов»

Тема: «Расстояния Левенштейна и Дamerau-Левенштейна»

Студент:

Пронин А. С.

Группа:

ИУ7-52Б

Преподаватель:

Власова Л. Л.

Оценка:

Москва

2021

Содержание

Введение	3
1 Аналитический раздел	4
2 Конструкторский раздел	7
3 Технологический раздел	8
4 Исследовательский раздел	9
Заключение	10
Список использованных источников	11

Введение

Цель работы – получить навык динамического программирования.

Задачи работы:

- изучить расстояния Левенштайна
- изучить расстояния Дамерау-Левенштайна
- разработать алгоритм вычисления расстояния:
 - Левенштейна обычным способом (матричным)
 - Левенштейна рекурсивным способом
 - Левенштейна рекурсивным способом с кэшированием
 - Дамерау-Левенштейна обычным способом (матричным)
- реализовать алгоритмы.
- провести сравнительный анализ процессорного времени выполнения реализации алгоритмов.
- провести анализ пикового значения затрачиваемой памяти в программе

1 Аналитический раздел

Расстояние Левенштейна

Расстояние Левенштейна (базовый вид редакторского расстояния) — это минимальное количество редакций необходимое для превращения одной строки в другую.

Редакторские операции бывают:

- I (insert) - вставка
- D (delete) - удаление
- R (replace) — замена

У этих трёх операций штраф = 1.

Еще одна операция:

- M (Match) — совпадение

Эта операция не имеет штрафа (он равен нулю).

Пусть s_1 и s_2 — две строки (длиной M и N соответственно) над некоторым алфавитом, тогда редакционное расстояние (расстояние Левенштейна) $d(s_1, s_2)$ можно подсчитать по следующей рекуррентной

формуле:

$$|x| = \begin{cases} 0, \text{ если } i = 0, j = 0; \\ i, \text{ если } i > 0, j = 0; \\ j, \text{ если } i = 0, j > 0; \\ \min \begin{pmatrix} D(s1[1 \dots i], s2[1 \dots j - 1]) + 1 \\ D(s1[1 \dots i - 1], s2[1 \dots j]) + 1 \\ D(s1[1 \dots i - 1], s2[1 \dots j - 1]) + \begin{bmatrix} 0, s1[i] == s2[j] \\ 1, \text{ иначе} \end{bmatrix} \end{pmatrix} \end{cases}$$

Расстояние Дамерау-Левенштейна

Вводится дополнительная операция: перестановка или транспозиция двух букв со штрафом 1. Если индексы позволяют и если две соседние буквы $s1[i] = s2[j - 1] \wedge s1[i - 1] = s2[j]$, то в минимум включается перестановка.

Пусть $s1$ и $s2$ — две строки (длиной M и N соответственно) над некоторым алфавитом, тогда редакционное расстояние (расстояние Дамерау-Левенштейна) $d(s1, s2)$ можно подсчитать по следующей рекуррентной

формуле:

$$|x| = \begin{cases} 0, \text{ если } i = 0, j = 0; \\ i, \text{ если } i > 0, j = 0; \\ j, \text{ если } i = 0, j > 0; \\ \min \left(\begin{array}{l} D(s1[1 \dots i], s2[1 \dots j - 1]) + 1 \\ D(s1[1 \dots i - 1], s2[1 \dots j]) + 1 \\ D(s1[1 \dots i - 1], s2[1 \dots j - 1]) + \begin{bmatrix} 0, s1[i] == s2[j] \\ 1, \text{ иначе} \end{bmatrix} \\ D(s1[0 \dots i - 2], s2[0 \dots j - 2]) + 1 \end{array} \right), \\ \text{если } i > 1, j > 1, s1[i - 1] = s2[j - 2], s1[i - 2] = s2[j - 1] \\ \min \left(\begin{array}{l} D(s1[1 \dots i], s2[1 \dots j - 1]) + 1 \\ D(s1[1 \dots i - 1], s2[1 \dots j]) + 1 \\ D(s1[1 \dots i - 1], s2[1 \dots j - 1]) + \begin{bmatrix} 0, s1[i] == s2[j] \\ 1, \text{ иначе} \end{bmatrix} \end{array} \right), \text{ иначе} \end{cases}$$

2 Конструкторский раздел

3 Технологический раздел

4 Исследовательский раздел

Заключение

Список использованных источников