Анализ Алгоритмов

Лабораторная работа №1

Юмаев Артур Русланович

ИУ7-55

Преподаватель: Волкова Л.Л.

Оглавление

[Введение 3](#_Toc18938329)

[Аналитическая часть 3](#_Toc18938330)

[Расстояние Левенштейна 3](#_Toc18938331)

[Расстояние Домерау-Левенштейна 5](#_Toc18938332)

[Конструкторская часть 5](#_Toc18938333)

[Технологическая часть 5](#_Toc18938334)

[Исследовательская часть 5](#_Toc18938335)

[Заключение 6](#_Toc18938336)

[Список литературы 6](#_Toc18938337)

# Введение

Данная работа посвящена исследованию “Редакторского алгоритма” или алгоритма Левенштейна и его дополнения – алгоритма Домерау-Левенштейна. Исследование является целью изучить математическую часть алгоритмов и запрограммировать их в нескольких вариантах (матричном и рекурсивном). Также будет проведена оценка алгоритмов по скорости работы и подготовлено их сравнение между собой.

# Аналитическая часть

Ставится задача поиска минимального расстояния D(S1, S2) между двумя словами S1 и S2. M = |S1|, N = |S2|.

Расстояние Левенштейна

Расстояние Левенштейна - расстояние между двумя строками в теории информации и компьютерной лингвистике — это минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую.

Для расстояние Левенштейна рассмотрим граничные случаи:

D(char, 0) = 1 // Delete

D(0, char) = 1 // Insert

D(0, 0) = 0 // Match,

Где D – расстояние,

char – любой символ

Расстояние Левенштейна определяется как: D(S1[1…L(S1)], S2[1…L[S2]])

Общий случай для рекурсивного подхода:

D(S1[1…i], S2[1…j]) = min(

D(S1[1…i], S­­2[1…j-1]) + 1,

D(S1[1…i-1], S­­2[1…j]) + 1,

D(S1[1…i-1], S­­2[1…j-1]) + {0 if S1[i] = S2[j] else 1}

)

Общий случай для матричного подхода:

A = (L(S1) + 1) x (L(S2) + 1) – матрица

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | … | L(S1) |
| 1 | … | … | … |
| … | … | … | … |
| L(S2) | … | … | … |

Для I > 0, j > 0:

|  |  |
| --- | --- |
| X | Y |
| Z | Q |

Q = min(

Y + 1,

Z + 1,

X + {0 if S1[i] = S2[j] else 1}

)

## Расстояние Домерау-Левенштейна

Алгоритм Домерау-Левенштейна заключается в том, что к аргументам функции min(…) добавляется еще одно условие:

+ {1 if S1[i] = S2[j - 1] & S2[j] = S1[i - 1] else 0}

# Конструкторская часть

# Технологическая часть

В качестве языка программирования был выбран Python из-за того, что для него существуют простые и быстрые библиотеки для работы с матрицами, которые упрощают процесс разработки. Библиотеки для работы с матрицами – Pandas и NumPy. Статистические данные отображаются с помощью библиотеки Matplotlib.

# Исследовательская часть

На рис. 1 приведено сравнение алгоритмов по времени работы относительно количества букв (от 1 до 9) в разных словах. По нему мы видим, что рекурсивные алгоритмы показали крайнюю неэффективность и рост времени работы для слов длинной больше 5. Алгоритмы, основанные на матричном подходе для слов менее 9 символом не показали значительное снижение эффективности по времени работы.

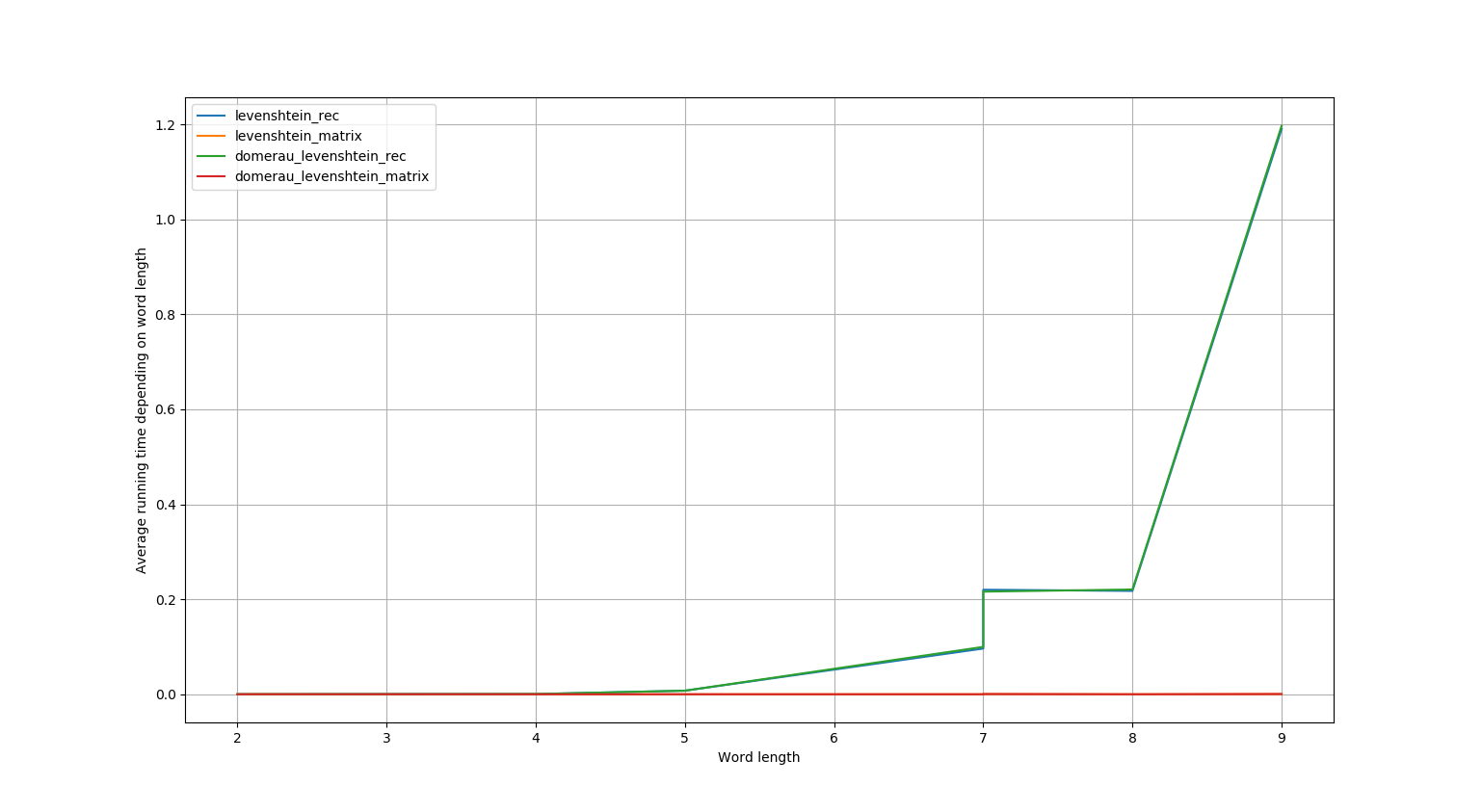


Рис. 1

# Заключение

# Список литературы