Лабораторная работа № 4 по курсу дискретного анализа: Строковые алгоритмы

Выполнил студент группы М80-208Б-22 *Ширяев Никита*.

Условие

Необходимо реализовать поиск одного образца в тексте с использованием алгоритма Z-блоков. Алфавит — строчные латинские буквы.

Метод решения

Программа использует алгоритм вычисления Z-функции для строки с разделителем, чтобы найти все вхождения заданного образца (подстроки) в строке. В данном случае программа ищет в строке, полученной путем объединения искомого образца и входной строки с использованием специального символа-разделителя `$`.

Разделитель выбирается таким образом, чтобы не принадлежать алфавиту исходного образца, чтобы избежать неправильного сопоставления вхождений образца с другими частями строки.

Алгоритм вычисления Z-функции позволяет эффективно определить длину наибольшего общего префикса (подстроки, начинающейся с первого символа строки) между текущей позицией и всеми возможными суффиксами строки. Значения Z-функции, равные длине образца, указывают на точные вхождения образца в строку.

В итоге программа выводит позиции, соответствующие точным вхождениям образца в строку, на основе вычисленных значений Z-функции.

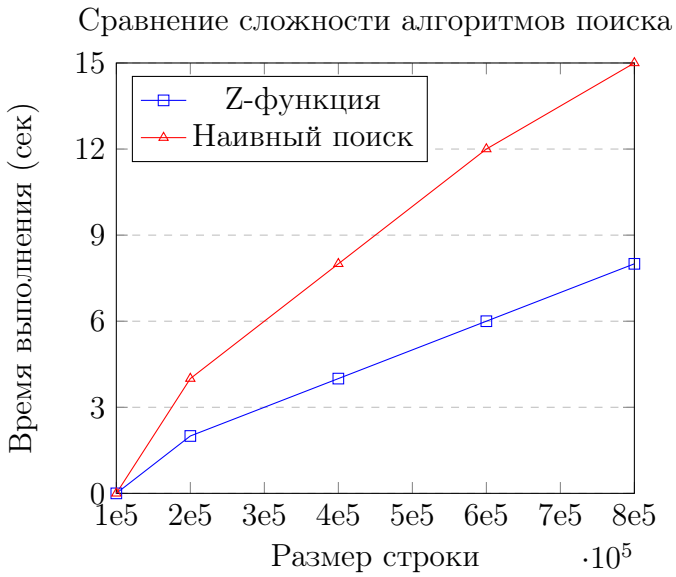
Описание программы

- Функция `computeZArray` вычисляет Z-функцию для входной строки. Значение Z[i] соответствует длине наибольшей подстроки, начинающейся с позиции i, которая также является префиксом всей строки.

- Функция `findPatternInText` строит новую строку, конкатенируя шаблон, специальный символ и исходный текст, и вычисляет Z-функцию для этой строки. Затем она просматривает полученный массив и выводит позиции, где Z[i] равно размеру шаблона, указывая на вхождения шаблона в исходном тексте.

- Функция `main` является точкой входа в программу. Она считывает текст и шаблон из стандартного ввода и вызывает функцию `findPatternInText`, чтобы найти все вхождения шаблона в текст.

Тест производительности



Выводы

Алгоритм Z-функции показал высокую эффективность по сравнению с наивным алгоритмом поиска. Из графика видно, что время выполнения алгоритма Z-функции остается стабильным и низким при увеличении размера входных данных. Это делает его привлекательным для использования в реальных приложениях, где размер входных данных может быть большим.

Наивный алгоритм поиска, несмотря на его простоту, не подходит для больших входных данных из-за его высокой сложности. Этот факт подтверждается на графике, где время выполнения наивного алгоритма увеличивается по мере роста размера входных данных.