ЛЕКЦИЯ 25
Z–функция строки и ее вычисление

Основной материал лекции взят с сайта.

1. Z–функция

1.1. Определение

Пусть дана строка s длины n. Тогда Z–функция от этой строки — это массив длины n, i-ый элемент которого равен наибольшему числу символов, начиная с позиции i, совпадающих с первыми символами строки s.

Иными словами, `z[i]` – это наибольший общий префикс строки s и её i–го суффикса.

Во избежание неопределённости, мы будем считать строку 0–индексированной — т.е. первый символ строки имеет индекс 0, а последний — $n-1$.

Первый элемент Z–функции, `z[0]`, обычно считают неопределённым. Мы будем считать, что он равен нулю.

1.2. Реализация

1.2.1. Тривиальный алгоритм

Программа №1.1. Тривиальный алгоритм

```
1    def z_function_trivial(s):
2        n = len(s)
3        z = [0]*n
4        i = 1
5        while i < n:
6            while i + z[i] < n and s[z[i]] == s[i+z[i]]: # Пока мы не дошли до
         конца и символ на позиции z[i] равен символу на рассматриваемой позиции + z[i]
         (т.е., это и есть основная проверка)
7                z[i] += 1
8            i += 1
9        return z
```

Сложность такого алгоритма $O(N^2)$.

1.2.2. Эффективный алгоритм

Чтобы получить эффективный алгоритм, будем вычислять значения `z[i]` по очереди — от i=1 до $n-1$, и при этом постараемся при вычислении очередного значения `z[i]` максимально использовать уже вычисленные значения.

Программа №1.2. Эффективная реализация Z–функции

```
1    def z_function(s):
2        z = [0]*len(s)
3        left = 0
```

```
4         right = 0
5         x = 0
6         for i in range(1, len(s)):
7             if i <= right:
8                 x = min(z[i-left], right - i + 1)
9             else:
10                x = 0
11            while i+x < len(s) and s[x] == s[i+x]:
12                x += 1
13            if i + x - 1  > right:
14                left, right = i, i + x - 1
15            z[i] = x
16        return z
```

Этот алгоритм выполняется за линейное время.

1.3. Применение

Применения Z–функции:

- Поиск подстроки в строке.
- Поиск количества различных подстрок в строке.
- Сжатие строки.

Запишем реализацию поиска количества различных подстрок в строке.

Программа №1.3. Поиск количества различных подстрок в строке

```
1     def count_different_substrings(s):
2         n = len(s)
3         start = s[0]
4         res = 0
5         for i in range(1, n):
6             t = start[::-1]
7             k = len(t) - max(z_function(t))
8             res += k
9             start += s[i]
10        return res
```

Г. С. Демьянов, VK
С. С. Клявинек, VK
А. С. Кожарин, VK