

Pattern matching

Алгоритмы и структуры данных поиска

Алексей Колесов

ШАД 2015. Минск

26 декабря, 2014

Содержание

- 1 z-алгоритм
 - Определения
 - Реализация

z-блок

Определение

Будем называть **z-блоком** пару (l, r) в случае если подстрока $s[l \dots r]$ совпадает с префиксом строки s .

Примеры z-блоков

a	b	a	c	a	b	a
0	1	2	3	4	5	6

Например, для строки «abacaba» z-блоками будут являться пары $(0, 5)$, $(4, 7)$, $(2, 3)$ и другие.

z-функция


Определение

z-функция $z[i]$ — длина максимального z-блока, который начинается в символе i .

Нулевой символ

Согласно определению, $z[0] = |s|$. Однако, для удобства реализации, рассуждений и причин, коих вам пока не дано понять, будем считать, что $z[0]$ равно котику.

Пример

a	b	a	c	a	b	a
	0	1	0	3	0	1

Лобовая реализация

```
1  vector<size_t> calculate_z(const string& s) {
2      vector<size_t> z(s.size());
3
4      size_t left = 0, right = 0;
5      for (size_t i = 1; i < z.size(); ++i) {
6          if (right <= i) {
7              left = right = i;
8              while(right < s.size() && s[right] == s[right-left]) ++right;
9              z[i] = right - left;
10         } else {
11             if (z[i - left] < right - i) {
12                 z[i] = z[i - left];
13                 continue;
14             }
15             z[i] = right - i;
16             while (i + z[i] < s.size() && s[i + z[i]] == s[z[i]]) {
17                 ++z[i];
18             }
19             if (i + z[i] > right) {
20                 left = i;
21                 right = i + z[i];
22             }
23         }
24     }
25     return z;
26 }
27 }
```

Листинг 1: Лобовая реализация Z-алгоритма

Чуть лучше

```
1  vector<size_t> calculate_z(const string& s) {  
2      vector<size_t> z(s.size());  
3  
4      size_t left = 0, right = 0;  
5      for (size_t i = 1; i < z.size(); ++i) {  
6          if (right <= i) {  
7              left = right = i;  
8              while(right < s.size() && s[right] == s[right-left]) ++right;  
9              z[i] = right - left;  
10         } else {  
11             z[i] = min(right - i, z[i - left]);  
12             while (i + z[i] < s.size() && s[i + z[i]] == s[z[i]]) {  
13                 ++z[i];  
14             }  
15             if (i + z[i] > right) {  
16                 left = i;  
17                 right = i + z[i];  
18             }  
19         }  
20     }  
21  
22     return z;  
23 }
```

Листинг 2: Чуть улучшенная реализация Z-алгоритма

Приемлимая версия

```
1  vector<size_t> calculate_z(const string& s) {  
2      vector<size_t> z(s.size());  
3  
4      size_t left = 0, right = 0;  
5      for (size_t i = 1; i < z.size(); ++i) {  
6          if (i < right) {  
7              z[i] = min(right - i, z[i - left]);  
8          }  
9          while (i + z[i] < s.size() && s[i + z[i]] == s[z[i]]) ++z[i];  
10         if (i + z[i] > right) {  
11             left = i;  
12             right = i + z[i];  
13         }  
14     }  
15  
16     return z;  
17 }
```

Листинг 3: Реализация Z-алгоритма