Поиск подстроки

Обозначения:

$T[1:n]$ - текст

$P[1:n]$ - шаблон

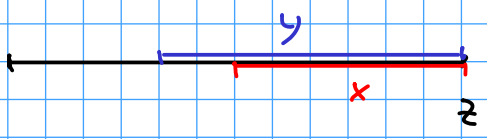
x, y - строки

$x \sqsupset y$ — x - суффикс y

$x \sqsubset y$ — x - префикс y

Утв:

$x \sqsupset z,\ y \sqsupset z \quad |x| \le |y| \Rightarrow x \sqsupset y$



Задача о поиске шаблона

Вход: T, P

Выход: 1. $\min i : P \sqsupset T[1:i+n-1]$

2. Все такие i

Наивный подход

$O(n \cdot m)$ $\quad T = aaa \ldots a$

$P = aaaab$

Алгоритм Карпа - Рабина

$O(n+m)$ в среднем

Z - функция

$Z[i]$ = длина max префикса T начинающегося в i

T = abcabacabacabc

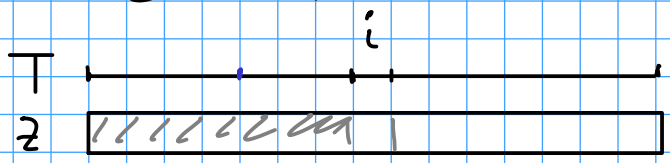
Z = 1200201020300

не встречается в P и T

Найдем $Z(P\#T)$ или решим обе задачи

] $z[1] \dots z[i-1]$ - посчитаны

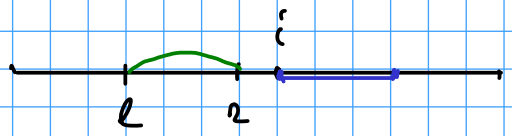
$z[i] = ?$



] $T[l, r]$ - самый правый префикс P, который мы до этого момента посчитали.

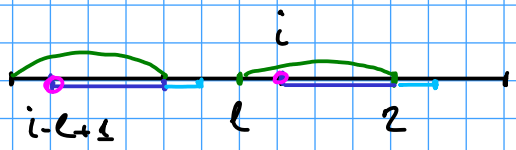
Т.е. $z[l] = |T[l, r]|$

①

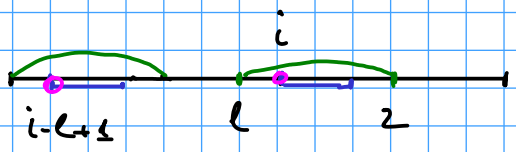


Мы вычисляем $z[i]$ наивно

②



$z[i] = z[i-l+1]$ + проверить далее



$z[i] = z[i-l+1]$

```
Z(T[1:m]):
    Z[i] = 0                                   O(m)
    Z[1] = m
    l = r = 0
    for  i = 2  to  m
        |  if  i ≤ r
        |      Z[i] = min(Z[i-l+1], m-i+1)
O(m)    |  while Z[i]+i ≤ m && T[Z[i]+1] = T[i+Z[i]]
        |      Z[i] = Z[i] + 1
        |  if  i + Z[i] - 1 > r
        |      l = i
        |      r = i + Z[i] - 1
```

⇒ Поиск шаблона (т.е. $Z(P\#T)$) работает за $O(m+n)$

Алгоритм Кнута - Морриса - Пратта

$\pi(P)$ - массив

≡ $\pi[i]$ - это $\max\ j<i : P[1:j] \sqsupset P[1:i]$

P = a b c a b a c a b a b c ≡ Префикс - функция

π = 0 0 0 1 2 1 0 1 2 1 2 3

Аналогично: $\pi(P\#T)$ то мы решаем обе задачи поиска

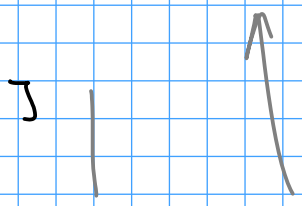
T = a b c a b c a b d P = a b c a b d

a b c a b d

→ a

→ a b c a b d

```
Prefix Function ( P[1:n] )
k = 0 // совпало символов                    O(n)
for i = 1 to n
O(n) | while k > 0 && P[k+1] ≠ P[i]
     |     k = π[k]
     | if P[k+1] = P[i]
     |     k = k+1
     | π[i] = k
```



k=5

abcabdabcabc

k=2

```
Search ( P[1:n], T[1:m] )
    k = 0 // совпало символов
O(n) π ← Prefix Function (P)                 O(n+m)
    for i = 1 to m
O(m) | while k>0 && P[k+1] ≠ T[i]
     |     k = π[k]
     | if P[k+1] = T[i]
     |     k = k+1
     | if k = n
     |     return i
```

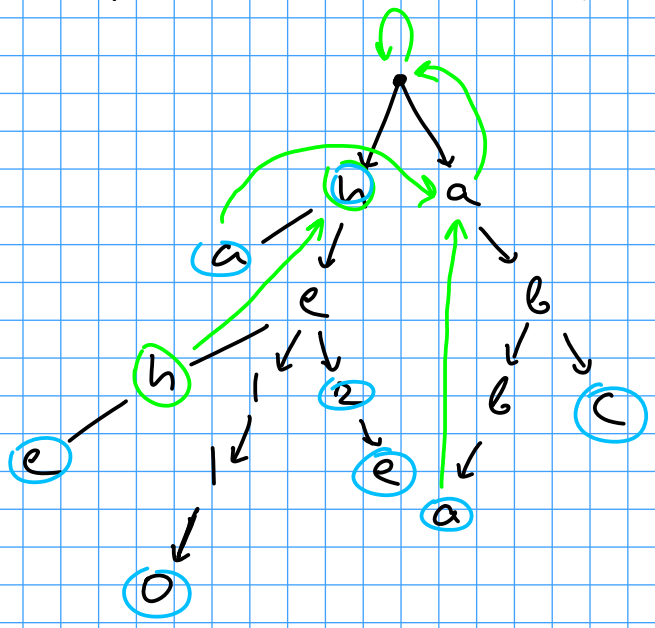
P = abcabacababc

π = 000121012123

Алгоритм Ахо-Корасик

Задача: множество $\{P_i\}$ – шаблоны
Найти все вхождения P_i в T

Бор $\{hello, hez, heze, abc, abba\}$
$h, ha, hehe\}$



~ конечному автомату

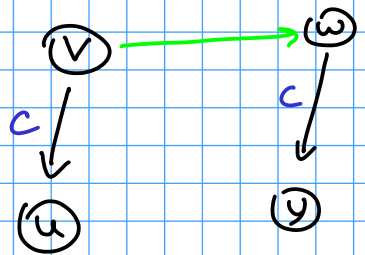
1. Строим бор из шаблонов

$O(\sum |P_i|)$ Память: $O(\sum |P_i| \cdot |A|)$

$= n$ ↑ алфавит

2. Восстанавливаем суффиксные ссылки

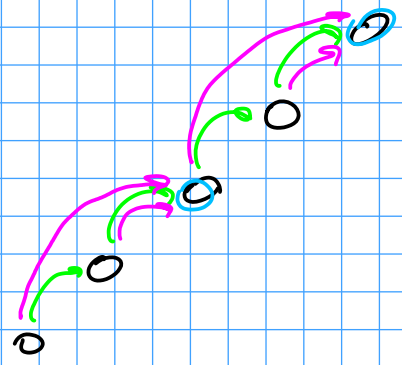
Поиск в ширину



$O(n)$

3. Для нахождения всех вхождений добавляются "сжатые ссылки"!
заканчиваются только в

по идее шаблона



$O(m)$

Сложность: $O(n) + O(m) + O(\#вхождений)$

$O(n)$ — построение бор-стрелки

$O(m)$ — прочитать текст

$$O(m + n + \#вхождений)$$