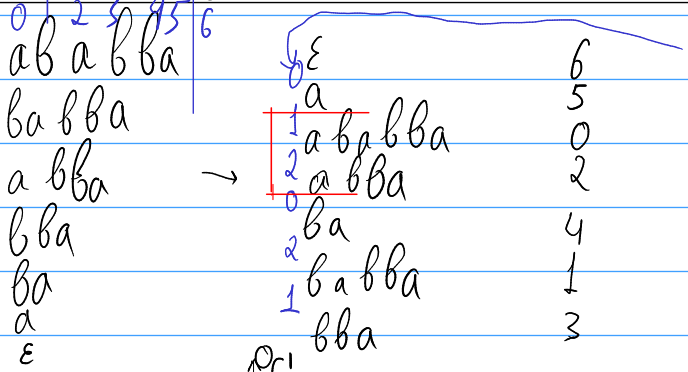
Суффиксный массив

Можно ли так запрепроцессить текст так, чтобы потом в нем можно было искать все, что захотим?

Как нам сохранить всевозможные подстроки текста, чтобы отвечать на разные запросы

Largest Common Prefix

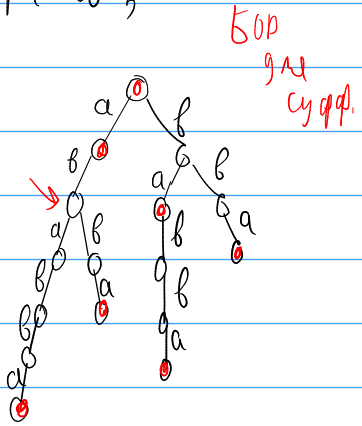
lcp - общий путь в боре

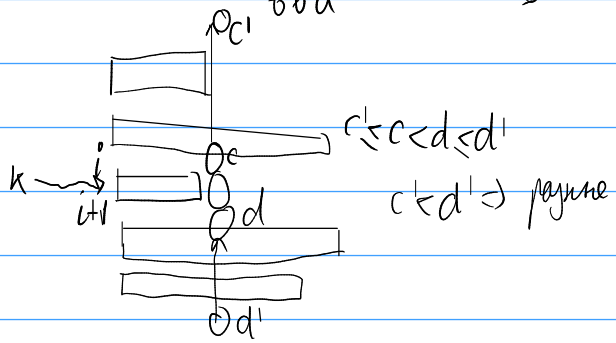


lcp соседних

(ав)

lcp("ав",





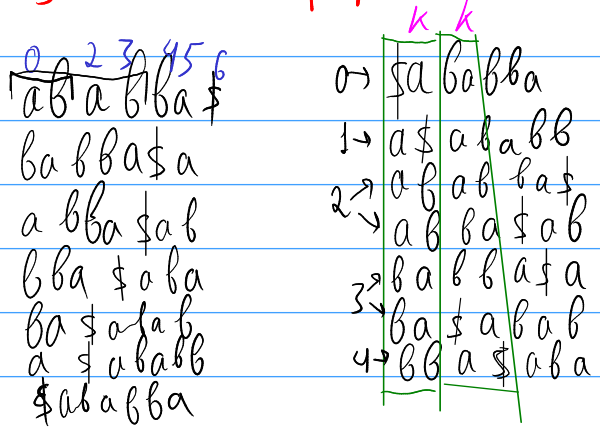
Критерий бинпоиска

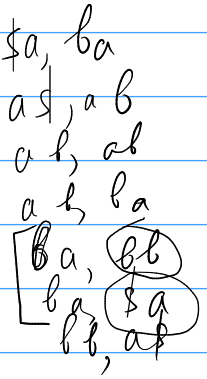
$s[a[v]...] \geq w$

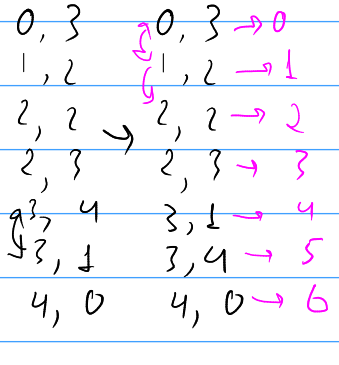
$s[a[x]...] > w$ и w не префикс

$O(|w| \log |t|)$

Надо научиться сортировать все суффиксы.







abaaba$

1 2 1 1 2 1 0 — ранги циклов

6 0 2 5 1 3 4 → порядок по первым буквам

(0,1)(1,2)(1,2)(1,0)(2,1)(2,2)(2,1)

(1,2) (2,1) (1,2) (2,2) (2,1) (1,0) (0,1)

6 5 0 2 1 4 3

abaaba$

2 3 2 4 3 1 0

сортировка подсчётом → p, c — переставила; ранги циклов (по индексу)

```
k=0
while 2^k ≤ n
    for i=0..n-1
        t[i] = ⟨c_i, c_(i+2^k) mod n⟩
    сортируем i по t[i] → p
    c_p[0] = 0
    for i=1..n-1
        if t[p[i]] = t[p[i-1]]
            c_p[i] = c_p[i-1]
        else
            c_p[i] = c_p[i-1] + 1
    k++
```

Я ничё не поняла. посмотрю лекцию

Пятая Лекция. Суффиксный массив. $O(n \log n)$

$S = abaabba$

Суффиксы в лексикографическом порядке:

6	a	
5	aabaabba	
0	abaabba	
2	abba	
4	baabba	
1	bba	
3	ba	
=	p	суффикс. массив

Quick sort строк

$O(n \log n \cdot comp)$

$A = a_1 a_2 a_3 \dots x$

$B = a_1 a_2 a_3 \dots y$ (n)

$A = xy$

$B = yy$

В среднем время работы компаратора $O(1)$ (Если строки рандомные)

Примем \$

$S = abaabba\$$

\$ – маркер конца строки

6	$	
5	a$	
0	abaabba$	
2	abba$	
4	baabba$	
1	bba$	
3	ba$	
=	p	2^k

Порядок не изменился

\$ всегда будет меньше => никогда не сравнивается

$s[i..n-1]$

↓

$s[i..i+n-1]$ (цикл)

↓

$k = 0..\lceil \log n \rceil$ $s[i..i+2^k-1]$

$k=0$

$s[i..i+2^k-1] = s[i..i]$

$O(n\log n)$

p	s[i..i]
0	a
1	b
2	a
3	b
4	b
5	a
6	$

sort → $O(n\log n)$

p	s[i..i]
6	$
0	a
2	a
5	a
1	b
3	b
4	b

Маленькая бонусная задача

$a[i] < a[j]$?

a: 1 5 8 8 13 13 13

$i<j \Rightarrow a[i] \le a[j]$

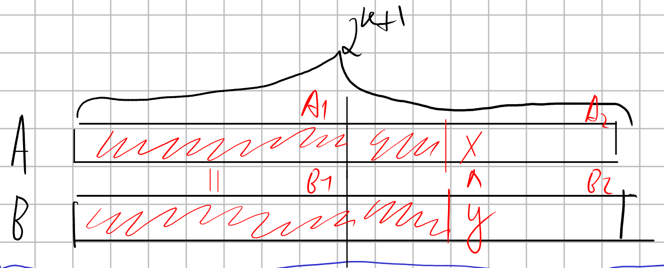
a: 1 5 8 8 13 13 13

c: [0 1 2 2 3 3 3]

$a[i] < a[j] \Leftrightarrow c[i] < c[j]$

Переход $k \longrightarrow k+1$

отсорт. $s[i..i+2^k-1] \longrightarrow$ отсорт. $s[i..i+2^{k+1}-1]$



$$A<B \Leftrightarrow (A_1<B_1) \vee (A_1=B_1 \wedge A_2<B_2)$$

сравнение строк 2^k

ababba$ (0 1 2 3 4 5 6)

k=0

p	s[i..i]	c
6	$	0
0	a	1
2	a	1
5	a	1
1	b	2
3	b	2
4	b	2

k=1

p		$c_1 c_2$
0	ab	12
1	ba	21
2	ab	12
3	bb	22
4	ba	21
5	a$	10
6	$a	01

sort →

p		c
6	01	0
5	10	1
0	12	2
2	12	2
1	21	3
4	21	3
3	22	4

k=2

p	
0	22
1	34
2	23
3	41
4	30
5	12
6	03

sort →

	p	c
03	6	0
12	5	1
22	0	2
23	2	3
30	4	4
34	1	5
41	3	6

Можно останавливаться, так как max c = 6

Сортируем пары (a_i, b_i)

$a_i, b_i \in [0 - n-1]$

Цифровая сортировка $O(n)$ ——— ~~log n~~ ——→ $O(n\log^2 n)$

Log n итераций

~~$O(n\log n)$~~

$O(n\log^2 n)$

$O(n\log n)$

S = ababba (0 1 2 3 4 5)

p:		
6	ε	
5	a	< a
0	ababba	≥ ab
2	abba	
4	ba	
1	babba	
3	bba	

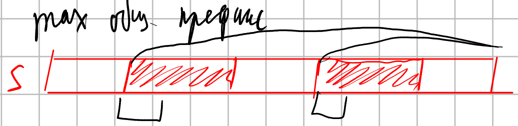
(вверху: ≥ ab)

T = ab |T| = m

бинпоиск

$O(\log n \cdot |m|)$

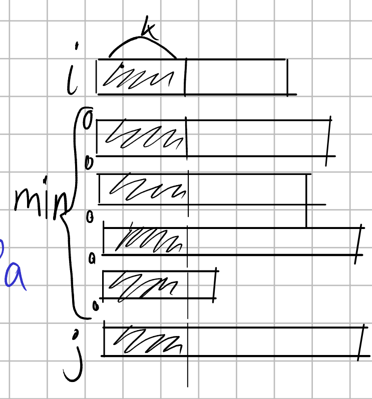
$lcp(s[i..n-1], s[j..n-1])$ — max общ. префикс



посчитаем lcp соседних

lcp	p:	
	6	ε
0	5	a
1	0	ababba
2	2	abba
0	4	ba
2	1	babba
1	3	bba

lcp(ba, babba, bba) = 0



Научимся находить lcp соседних

lcp: p:

	6	ε
	5	a
1	0	ававва
	2	авва
	4	ва
	1	вавва
	3	вва

← самый длинный

lcp: p:

	6	ε
	5	a
1	0	ававва
	2	авва
	4	ва
2	1	вавва
	3	вва

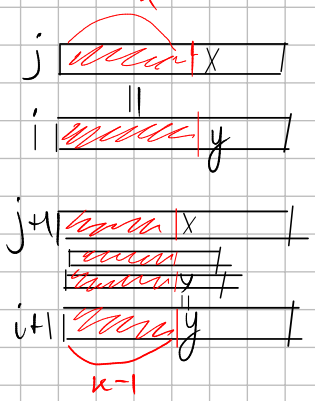
lcp: p:

	6	ε
	5	a
1	0	ававва
2	2	авва
	4	ва
2	1	вавва
	3	вва

4	ва...	
1	ва...	
5	а...	
	а	
2	а...	

lcp: p:

0	6	ε
	5	a
1	0	ававва
2	2	авва
0	4	ва
2	1	вавва
1	3	вва



k - число одинаковых символов

= k++ ≤ 2n раз

≠ след. суфф. k-- n раз

k ≤ n

O(n)