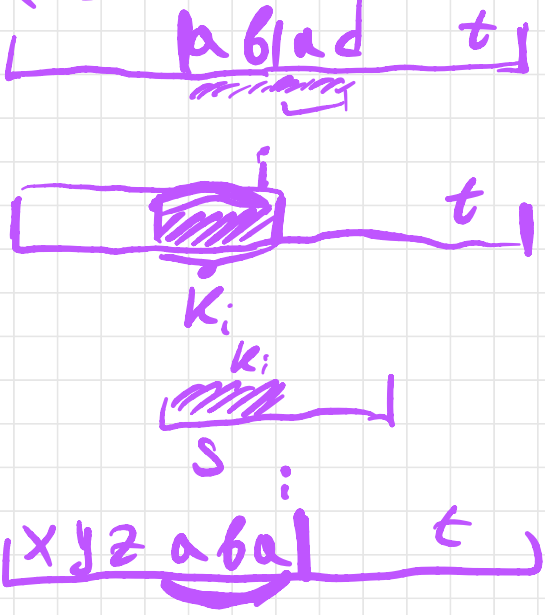
Aho-Corasick

Knuth-Morris-Pratt 1977

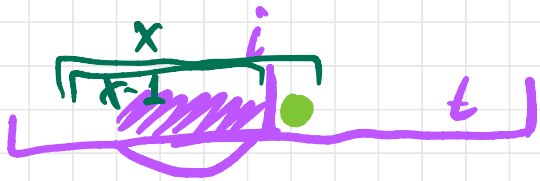
Z, π: $O(n+m)$ времени и памяти

КМП: $O(n+m)$ времени $O(n)$ памяти

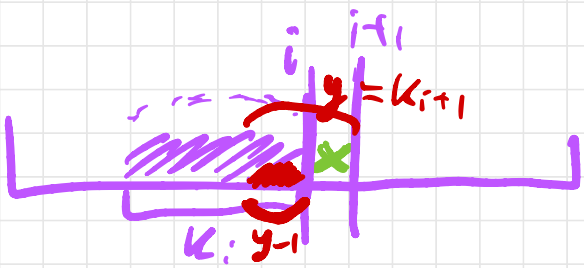
abacaba$



$k_i = h$ → вхождение.



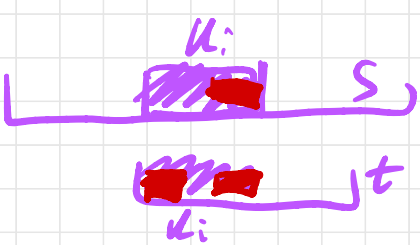
k_i $\qquad$ $k_{i+1} = k_i + 1$

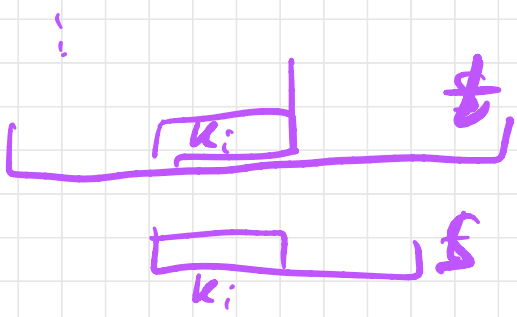


$k_{i+1} = y$, по

$y-1$ зва кандидат на k_i

- k_i
- $\pi[k_i - 1]$
- $\pi[\pi[\ ] - 1]$





$k_{i+1} \leftarrow (k_i) + 1$ если след. символ совпал.

$k_{i+1} \leftarrow \pi[k_i - 1] + 1$ если не совпал

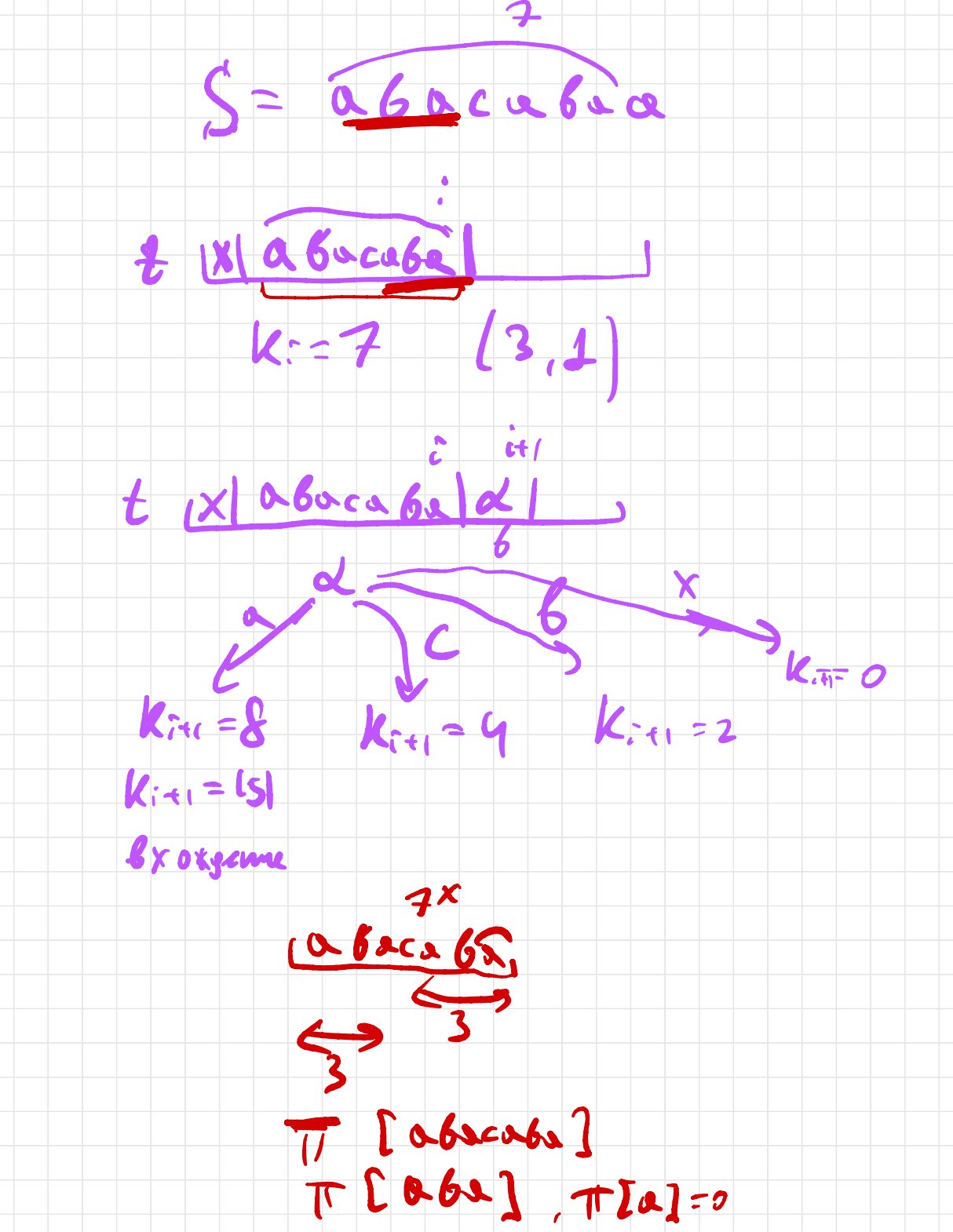
↓

...

Алгоритм.

1) вычислить $\pi[\$]$

2) идти по символам t и поддерживать текущее k_i


S = abacabaa
7
t |x| abacaba |
k_i = 7 (3,1)
i
i+1
t |x| abacaba | a |
b
a
c
b
x
k_{i+1} = 0
k_{i+1} = 8
k_{i+1} = 4
k_{i+1} = 2
k_{i+1} = 5
Выходим
7x
abacaba
3
3
π [abacaba]
π [aba] , π [a] = 0

$k_i \longrightarrow k_i + 1$

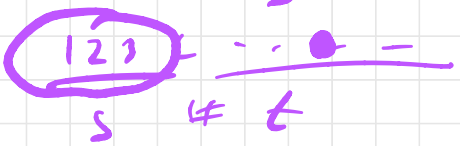
$\downarrow$

$\pi[k_i - 1] \rightarrow \pi[k_i - 1] + 1$

$\downarrow$

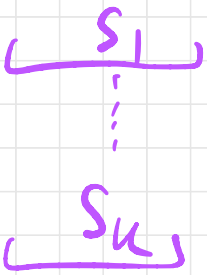
$\pi[\pi[\ldots] - 1]$

$\pi[s \# t]$



$\pi[\bullet] = \pi[s_0 \ldots s_{i-1}]$

Aho - Corasick 1975

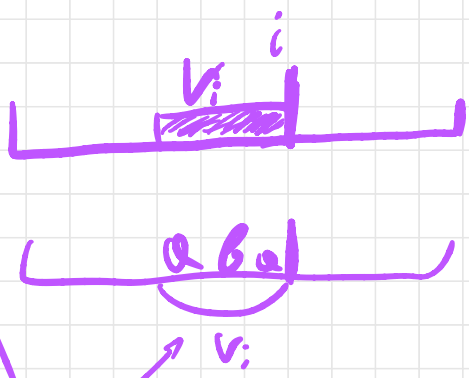


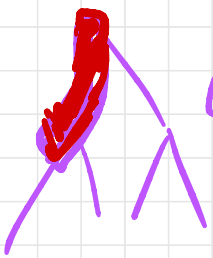
→ ? ∃ вхождение хотя кого-то из s_i

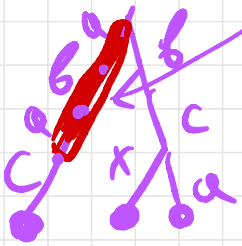
? сколько вхождений в сумме

? выписать все вхождения

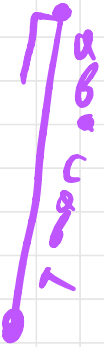
trie $(s_1 \dots s_k)$

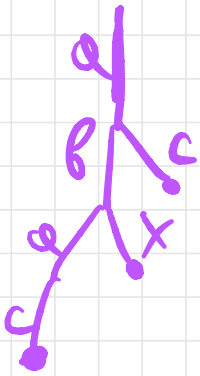

i
V;
a b a
V;

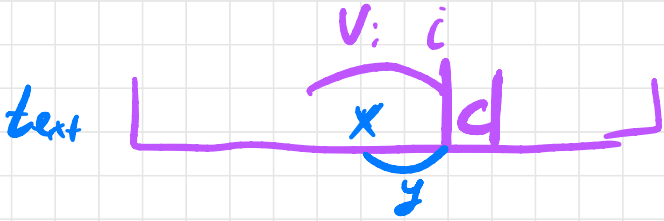



a
b
b
c
c
x
a

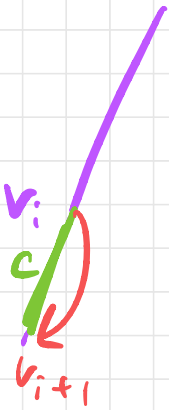

a
b
a

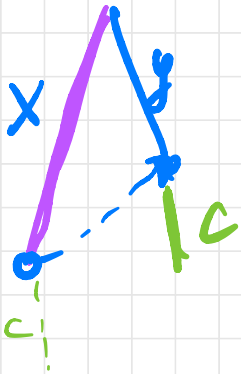

a
b
c
a
b

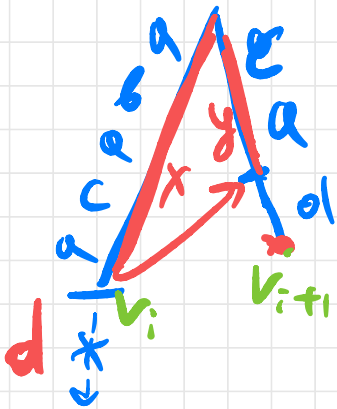

a
b
c
a
x
c


v_i
c
text
x
d
y

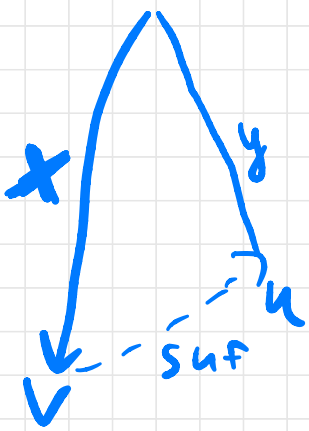

v_i
x
y

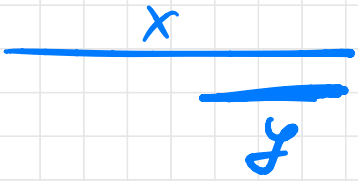

v_i
c
v_{i+1}


x
y
c
c


a
e
a
e
y
a
x
c
a
d
v_i
v_{i+1}

Определение
Суф-ссылки.

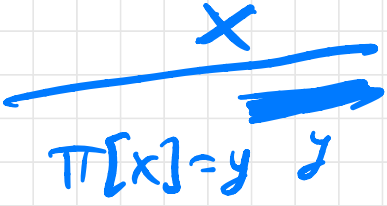


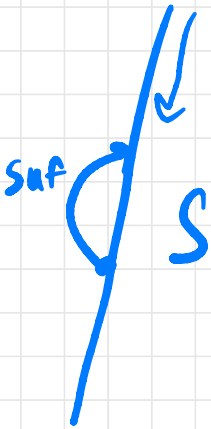


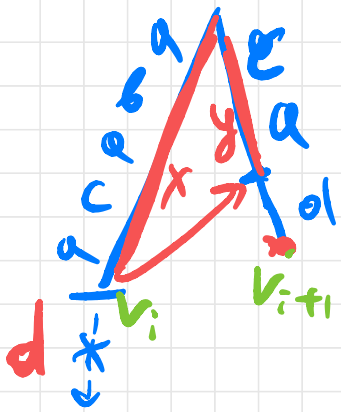
$v.suf = u$:

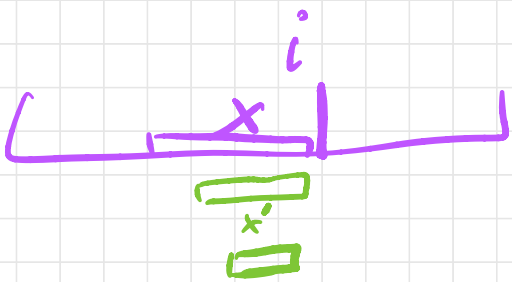
y – suffix x

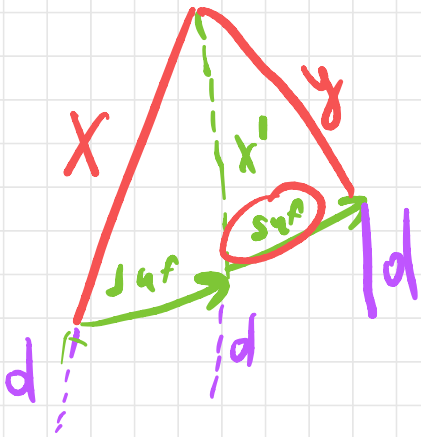
$|y| \to \max$



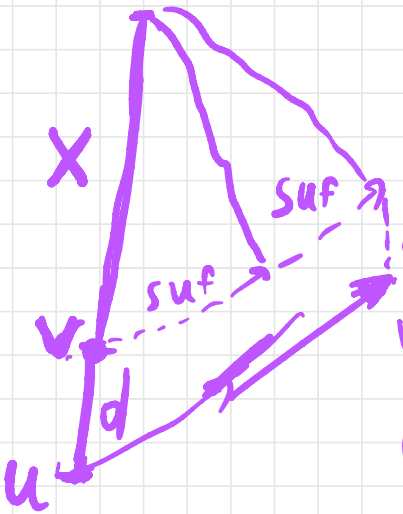



q
q
q
q
q
q
y
x
d
d
v_i
v_{i+1}


i
x
x'


X
y
x'
suf
suf
d
d
d

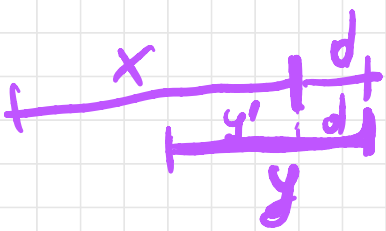
Как найти суф-ссылки



v.suf уже известно

u.suf неизвестно

u.suf = ?



y' - suffix x

```
class Node:
    def __init__(self):
        self.go = dict()
        self.suf = None
        self.term = False
```

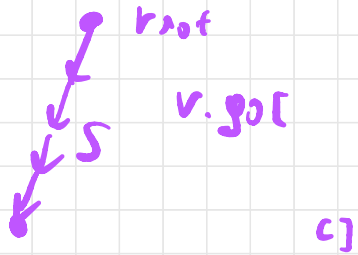
Struct Node {

go: dict(char, Node*);

suf: Node*

term: bool

}



root - корень бора

Алгоритм построения суф. ссылок

root.suf = None

q = Queue()

q ← root

while q: ← O(|S|)

v ← q // v.suf уже найден

v.go[c] = u

for (c, u) in v.go.items():

t = v.suf

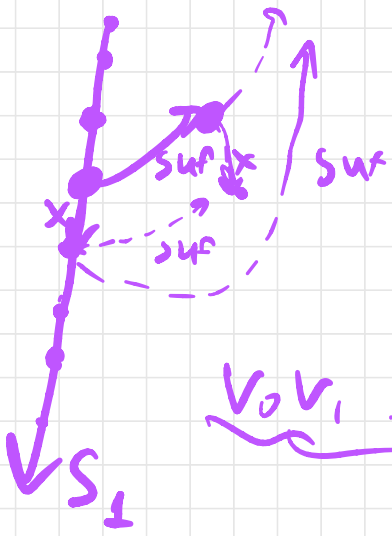
while t ≠ None and c ∉ t.go:

t = t.suf

u.suf = { root если t == None
t.go[c] иначе

q ← u

Время работы



$O(|S_1|)$

V_i - suf

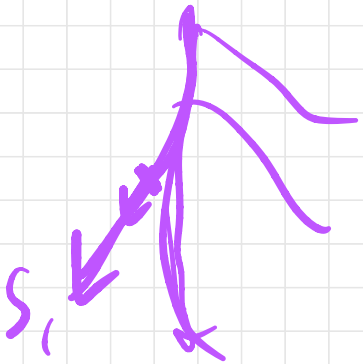
$V_0 \to V_1 \to \dots V_{|S_1|}$

V_0.suf V_1-suf

k_0 k_1 $k_{|S_1|}$

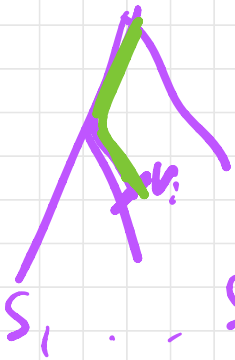
$k_0 = 0$ $k_1 \leq k_0 + 1$ $k_{|S_1|} \leq |S_1|$

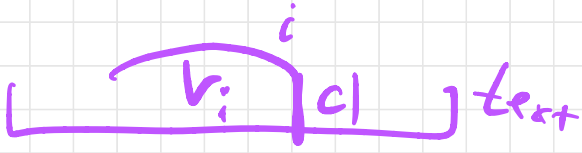
k_1 может быть и меньше

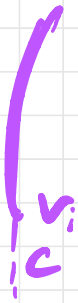


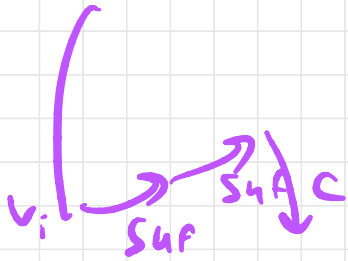
$O(\Sigma |S_i|)$

Ахо-Корасик



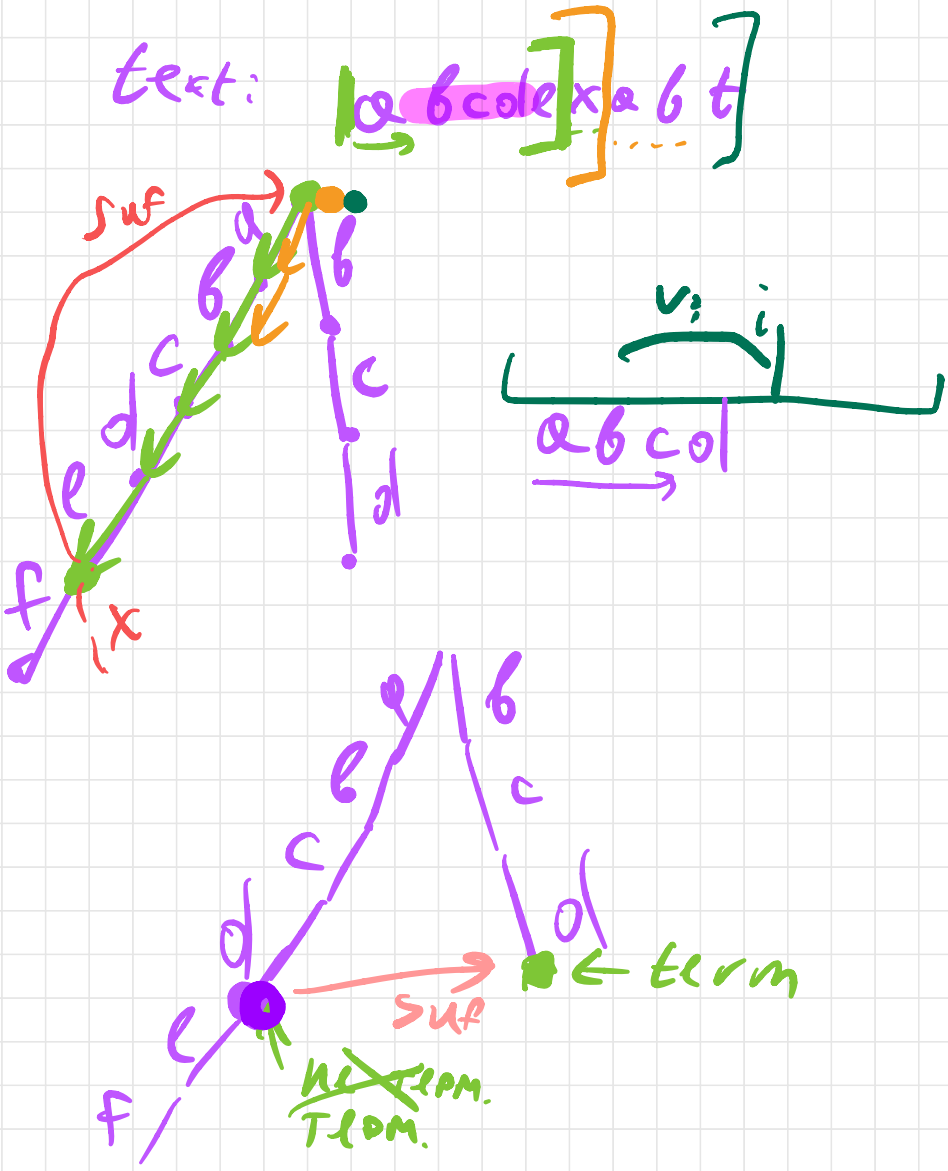






s_1 a b c d e f

s_2 = b c d

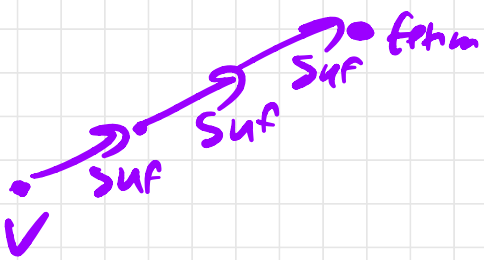


a b c d

1) Если мы хотим найти хотя
бы какое-то вхождение
/ проверить что оно
существ.

$$V.term \mid= V.suf.term$$

6 Алгоритм построения

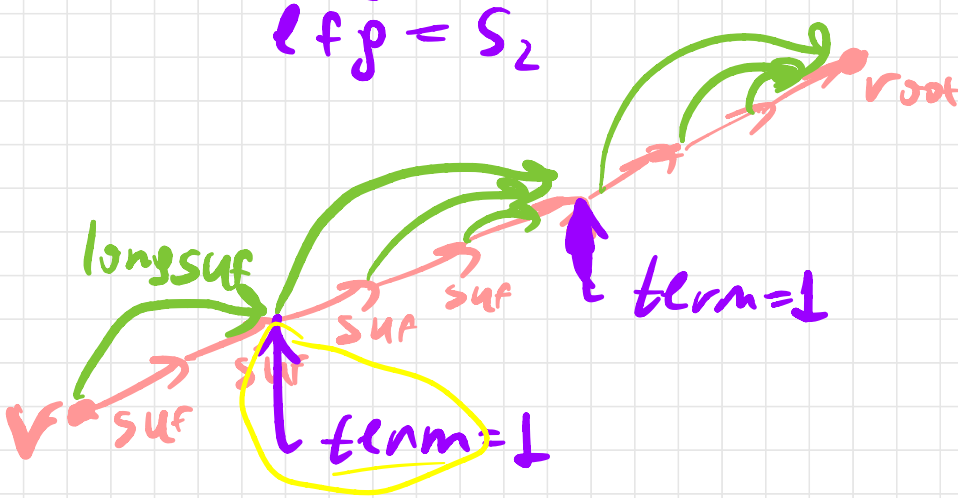


2) Что мы хотим выполнить
все вхождения.

aBcdefg

cdefg = S_1

efg = S_2



$O(\Sigma|S_i| + k)$

↑ число входящий