

# Суффиксные массивы

# Суффиксные массивы

Пусть задан текст  $T$  длины  $m$ .

Нужно так подготовить текст  $T$ , чтобы за минимальное время находить вхождения образца  $P$  длины  $n$  в текст  $T$

# Суффиксные массивы

- . В 1993 году Манбер (Manber U.) и Майерс (Myers G.) предложили для решения задачи о подстроке структуру, названную **суффиксным массивом**, которая достаточно рационально использует память и работает почти так же быстро, как суффиксные деревья ( $O(n)$ )

# Суффиксные массивы.

Пусть задана  $m$ -символьная строка  $T$ .

**Суффиксным массивом** для  $T$ , обозначенным  $Pos$ , называется массив целых чисел от 1 до  $m$ , определяющих лексикографический порядок всех  $m$  суффиксов строки  $T$ .

# Пример суффиксов и суффиксного массива для строки «абракадабра».

1	абракадабра	$Pos[0]=11$	
2	бракадабра	$Pos[1]=8$	а
3	ракадабра	$Pos[2]=1$	абра
4	акадабра	$Pos[3]=6$	абракадабра
5	кадабра	$Pos[4]=4$	адабра
6	адабра	$Pos[5]=9$	бра
7	дабра	$Pos[6]=2$	бракадабра
8	абра	$Pos[7]=7$	дабра
9	бра	$Pos[8]=5$	кадабра
0	ра	$Pos[9]=10$	ра
11	а	$Pos[10]=3$	ракадабра

# Суффиксный массив

Суффиксный массиве  $Pos$  не занимает много памяти.

Огромный плюс суффиксных массивов — их размер в памяти определяется только размерами текста  $T$  и никак не зависит от его алфавита.

Суффиксный массив можно использовать для поиска всех вхождений в  $T$  образца  $P$  за  $O(n + \log_2 m)$ .

# Построение суффиксного массива

Упорядочим суффиксы по первой букве и занесём результат в  $Pos$ .

**Корзиной** будем называть несколько соседних суффиксов с одинаковыми первыми буквами.

Сделаем разбиение на корзины мельче, пока количество корзин не совпадёт с длиной  $m$  строки  $T$ .

# Построение суффиксного массива

Последний суффикс (он же — последний символ строки  $T$ ) перенесём на первое место в своей корзине.

Далее сортируем в каждой корзине суффиксы по второму символу.

Обновляем разбиение на корзины: в каждой суффиксы, совпадающие по двум символам

Продолжаем процесс до получения полностью массива



# помощью суффиксного массива

Если образец  $P$  входит в строку  $T$ , то он является префиксом какого-нибудь суффикса  $T$ .

. При этом все вхождения  $P$  в  $T$ , если они есть, в суффиксном массиве  $Pos$  будут находиться рядом.

Пример: образец «бра» находится в строке «абракадабра» начиная со второго и с девятого символа. «бра» — префикс 2-го и 9-го суффиксов слова «абракадабра», которые в суффиксном массиве  $Pos$  находятся рядом.

# Поиск образца в строке с помощью суффиксного массива

Вхождения  $P$  в  $T$  находим двоичным поиском в упорядоченном массиве.

Проверяем  $Pos[m / 2]$ . Если суффикс  $Pos[m / 2]$  лексикографически меньше, то первая позиция, где  $P$  входит в  $T$ , должна быть в первой половине  $Pos$ . Если суффикс  $Pos[m / 2]$  лексикографически больше, чем  $P$ , то первая позиция, где  $P$  входит в  $T$ , должна быть во второй половине  $Pos$ . Далее аналогично ищем  $P$  в половине массива  $Pos$ . И так далее, пока не найдём (если такие существуют) наименьший и наибольший индексы  $i_{\min}$  и  $i_{\max}$  такие, что образец  $P$  входит в текст  $T$  в позициях  $Pos[i_{\min}]$ ,  $Pos[i_{\min} + 1]$ , ...,  $Pos[i_{\max}]$ .

# Поиск образца в строке с помощью суффиксного массива

При использовании двоичного поиска в массиве  $Pos$  все вхождения образца  $P$  в текст  $T$  могут быть найдены за время  $O(n \log t)$ .

Для случайных строк метод работает за ожидаемое время, но на случай, если в  $T$  есть много длинных префиксов  $P$ , метод можно улучшить

# Поиск образца в строке с помощью суффиксного массива

*Простой ускоритель  $mlr$*

При двоичном поиске обозначим левую и правую границы текущего интервала поиска как  $L$  и  $R$ .

В начале работы поиска  $L = 1, R = m$ .

На каждой итерации лексикографически сравнивается образец  $P$  с суффиксом  $Pos[(L+R) / 2]$ .

Обозначим через  $l$  длину общего префикса  $Pos[L]$  и  $P$ ,  
через  $r$  — длину общего префикса  $Pos[R]$  и  $P$ , а  
через  $mlr$  — минимум из  $l$  и  $r$ .

Значение  $mlr$  приходится поддерживать при  
двоичном поиске.

Зная значение  $mlr$ , мы можем ускорить  
лексикографическое сравнение  $P$  с  $Pos[(L+R)/2]$ , так  
как для любого

числа  $i$  от  $L$  до  $R$  первые  $mlr$  символов в  $Pos[i]$   
одинаковы и сравнение можно начинать не с  
начала, а с позиции  $mlr+1$ . Наихудшее время  
остаётся прежним —  $O(n \log m)$ , однако Майерс и  
Манбер сообщают, что использование  $mlr$  даёт  
 $O(n + \log m)$ .