Студент: Маргарита Чудова
Группа: 2
Дата: 27 апреля 2022 г.

---

1. Проблема заключается в том, что в цикле for мы начинаем с 0, а не с 1.
   В цикле while мы лишних $n-1$ раз прибавляем к k единичку, потому что $s[i+k] == s[k]$ гарантированно выполняется при $i=0$. Тогда по завершении $r=n-1, l=0$.
   В следующих for будет $k=0$, так как $l=0$, а элемент $z[i-l]=z[i]$ еще пока будет нулем. Получается, подстроки сравниваются с самого начала в любом случае. Если строка состоит из одного и того же символа, то сравниваться они будут от начала до конца (потому что условие $s[i] == s[i+k]$ в while не срабатывает). На это уйдет

$$n + (n-1) + \ldots + 1 = \Omega(n^2)$$

4. Воспользуемся решением задачи 4 с семинара. Сделав предподсчет хешей за $\mathcal{O}(n)$ мы сумеем сравнить любые две строки за $\mathcal{O}(\log n)$.
   Вспомним алгоритм поиска k-й статистики — он работал за $\mathcal{O}(n)$, при этом сравнение элементов занимало $\mathcal{O}(1)$. Применим этот алгоритм, но к строкам — изменится только время сравнения, и из-за него асимптотика будет $\mathcal{O}(n \log n)$

5. Посчитаем хеши для всех префиксов s и t, и дальше все будем сравнивать через них
   Далее пройдемся по строке t, будем в ней искать подстроку а-ля s. Для этого для каждого i найдем наибольший общий префикс, и от конца этого префикса будем смотреть на остаток строки (то есть оставшееся до конца $t[i : i+|s|)$). Действуем следующим образом: смотрим на остаток строки, если он не совпадает с концом строки s, то смотрим на первый их несовпадающий элемент (его можем найти за $\mathcal{O}(\log |s|)$), сравниваем его с нужным элементом в s, если они равны, сдвигаемся на один элемент. Если набрали k различий, то прерываемся — условия задачи не удовлетворены. Итого как бы 3 цикла: один по t, другой по k, третий от 0 до $log|s|$ — укладываемся в нужную асимптотику.

6. Сделаем бинпоиск по $l = 1, \ldots, n$. Пусть взяли какое-то l, рассматриваем строки длины l. Будем делать предподсчет хешей таких строк и складывать их в хеш-таблицу, при этом будем хранить не только строки, но и левую и правую границу, то есть храним кортеж $(s[i,l], l_i, r_i)$ для каждой строки. На это у нас уйдет $\mathcal{O}(n)$. Пусть в хеш-таблице в какой-то ячейке есть два значения, для которых выполнено условие

$$r_1 < l_2 \ or \ r_2 < l_1$$

   Тогда получается, что хеши двух строк совпали, при этом строки не пересекаются. В таком случае идем бинпоиском по l вправо, если нет — влево.
   Итого бинпоиск у нас займет $\mathcal{O}(\log n)$, а на каждом его шаге происходит подсчет за $\mathcal{O}(n)$.

7. (a) Предпосчет: посчитать все хеши префиксов s и $s[:: -1]$, $\mathcal{O}(n)$
   Рассмотрим строки $s[l..r)$ и $s[l,r)[:: -1]$ (то есть перевернутую). Посчитаем их хеши ($\mathcal{O}(1)$), если они равны, то и строки равны, а значит $s[l..r)$ является палиндромом.

   (b) Предпосчет: посчитать все хеши префиксов s и $s[:: -1]$, $\mathcal{O}(n)$.
   Рассмотрим некоторую позицию i в строке. Идея такая: если у нас есть некоторый максимальный палиндром, симметричный относительно i, и его длина равна l, то всего симметричных относительно i палиндромов l. Поэтому чтобы найти все палиндромы, нам надо найти этот максимальный палиндром, узнать его длину и прибавить ее к общему счетчику, а потом перейти к следующему $i+1$.

Как это сделать для конкретного i? Выберем минимум из i и $n-i$ (то есть расстояние от элемента i до ближайшего края строки), и в зависимости от того, что мы выбрали, будем делать бинпоиск либо по $l = 0, \ldots, i$, либо по $l = i, \ldots, n$. Сначала рассмотрим строки четной длины: на каждом шаге сравниваем строки $s[i-l-1, \ldots, i)$ и $s[i, i+l+1)[:: -1]$ (за счет предподсчета хешей делаем это за $\mathcal{O}(1)$). Если строки равны, увеличиваем l, если нет — уменьшаем.
Аналогично рассмотрим строки нечетной длины: там уже не получится такой симметрии, но мы можем выбрать, по какую сторону от i строчка будет больше, тогда все такие палиндромы относятся только к одному i.
Чтобы было понятнее, рассмотрим пример: строка aabaa. Пусть i = 2, (то есть элемент b), и мы всегда сравниваем строки $s[i-l, i)$, $s[i+1, i+l+1)[:: -1]$, здесь $l = 2$. Таким образом, символ b мы не учитываем, но учитываем равные по обе стороны от него строки

Итого проходимся бинарным поиском за $\mathcal{O}(\log n)$ n раз, асимптотика $\mathcal{O}(n \log n)$