Домашняя работа

Котов Артем, МОиАД2020

28 ноября 2020 г.

Содержание

Is it time to lakonichnyi solutions? — Well…it's not

Task 1: Префиксы префиксов

Условие: Для каждого префикса строки найти количество его префиксов, равных его суффиксу $\mathcal{O}(n)$.

Решение.

Ох опять промазал :)

Рассмотрим подстроку, соответствующая наибольшему значению в префикс-функции:

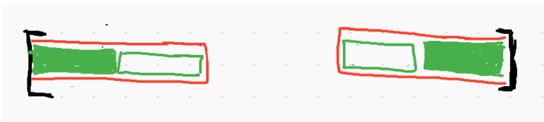

Рис. 1: Подстрока с наибольшим значением префикс-функции.

На этом рисунке черным обозначен как раз такой префик, у которого длина совпадающего префикса с суффиксом наибольшая (ох какая сложная семантика) среди всех префиксов исходной строки. Красным обозначены как раз соответствующие совпадающие кусочки.

Теперь заметим, что закрашенный зеленый кусочек — префикс красной, причем такой, что у него есть парный суффикс, т. е. в правом красном справа есть еще такой же закрашенный зеленый кусочек. Заметим далее, что так как левый красный равен правому красному, то правый закрашенный равен незакрашенному зеленому в левом красном, что даст еще один префикс, у которого будет нужное свойство.

Пусть $\pi[i]$ — префиксная функция. Заведем массив $d[i]$ — количество совпадений в соответствующих префиксах, тогда можно посчитать $d[i] = 1 + d[\pi[i]]$, т.е. получилась такая некая динамика, причем пройдемся мы один раз по всему массиву, ну и, видимо, для чистоты стоит сказать, что в крайнем самом левом случае положить d равным 1.

Замечание. *Вроде как, если похитрить, то можно посчитывать d в процессе обхода слева-направо, пока мы считаем префиксную функцию, тогда явно будет $\mathcal{O}(n)$*

■

Task 2: Почти равенство

Решение.

a)

■

Task 3: Из Z в префикс

Условие: Преобразовать Z-функцию в префикс-функцию без промежуточного восстановления строки. $\mathcal{O}(n)$.

Решение.

Переделанное решение, так как не все в этом мире так просто, я не смог сразу осознать, что они "растут" в разные стороны, хех. В целом, все так же идем по элементам $Z[i]$, пусть в ячейке хранится k, смотрим все также в элемент $\pi[i+k-1]$. Стоит заметить, что есть отношение связь между Z и π, а именно, что $\pi[i+j] \geq j+1$, так как по сути это просто длина этого сегмента.

Присваиваем $\pi[i+k-1] = j+1$, и так для всех элементов от $i+k-1$ до i, где $j+1$ длина соответствующего сегмента. Но тут может быть проблема, что мы перезапишем что-то, что уже было присвоено. Посмотрим, почему так делать не стоит: пусть мы куда-то присвоили x, пока были на позиции i, и пытаемся присвоить значение y с другой позиции $j > i$. Тогда заметим, что если мы присваиваем туда же, то $i + x = j + y$, ну тогда $y < x$, значит мы уменьшаем значение, которое было бы в префикс-функции. Таким образом, мы должны будем прерваться, если наткнулись на элемент, в котором уже что-то записано.

Пока что получилось, что у нас есть цикл внешний, который бежит по всему массиву, а внутренний цикл, кажется, что будет $\mathcal{O}(n^2)$, но это не так, так как на самом деле мы запишем в какую-то ячейку не более одного раза, последующие циклы будут прерываться сразу на этом элементе, конечно, это тоже стоит каких-то денег, но мы в итоге мы присваиваем всего лишь n элементов (это можно представить себе, что если мы закинули удочку, то следующее забрасывание будет проходить как бы из той точки, до куда мы забросили удочку в прошлый раз), при этом прерываний у нас будет не больше, чем n, то есть сложность будет что-то типа $\mathcal{O}(n) + n \cdot \mathcal{O}(1) = \mathcal{O}(n)$. ■

Task 4: Подпалиндромы

Условие: Найти количество подпалиндромов строки. $\mathcal{O}(n \log n)$

Решение.

Сделаем следующий предподсчет: заведем два как бы разных, но похожим хеша, который я обзову левый и правый хеши:

- Левый хеш будет хешировать префиксы слева-направо, им мы прохешируем все префиксы исходной строки
- Правй хеш будет хешировать суффиксы справа-налево, им мы прохешируем все суффиксы исходной строки

Так как палиндром — это нечто, что читается слева-направо так же, как и справа-налево, то относительно “центра симметрии”, если бы мы прохешировали левую и правую части в соответствии с приведенной выше схемой. Это будет стоить нам $\mathcal{O}(n)$

Теперь, займемся пока без деталей поиском таких палиндромов с помощью хешей: начнем с более понятного, как по мне, случая, когда мы рассматриваем палиндромы нечетной длины, чтобы “центр симметрии” был хорошо определен в виде некоторого элемента x. “Закинем удочку” от элемента x как можно дальше (в смысле так далеко, насколько позволят границы исходной строки в обе стороны). Получится некий симметричный отрезок с центром в точке x. Теперь, благодаря знанию соответствующих хешей на этих отрезках (так как знаем хеши на префикса/суффиксах, то за $\mathcal{O}(1)$ можем вычислять хеш на произвольном отрезке) мы можем быстро отвечать через сравнение хешей левого части с правой на вопрос действительно ли они симметричны. Пока мы научились только проверять, но не искать. Теперь заведем бинпоиск на самый длинный палиндром с центром в символе x, сдвигая в правильную сторону границы симметричного отрезка в зависимости от ответа на предыдущий вопрос. Эта процедура стоила нам только что $\mathcal{O}(\log n)$. Найдя самый длинный палиндром, по его длине (очень похоже на первую задачку, так как если откусить и слева, и справа по символу, то палиндром останется палиндромом), найдем количество палиндромов с центром в x. Так пробежимся по всем x, в итоге все удовольствие стоило нам $\mathcal{O}(n \log n + n) = \mathcal{O}(n \log n)$.

Закономерно задать вопрос: а что делать если у нас четные палиндромы? Тут поступим аналогичным образом, только будем рассматривать уже пары соседних xx.

■