Первый курс, весенний семестр
Практика по алгоритмам #11
Строки: суффиксные структуры

Contents

Новые задачи

1. **Поиск подматрицы.**
   Даны две матрицы чисел A и B.
   Нужно проверить, является ли B подпрямоугольником A за $\mathcal{O}(|A| + |B|)$.
2. **Хитрый поиск.**
   Дан словарь, постройте структуру, чтобы быстро отвечать на запросы
   (а) `get(s)` – количество строк в словаре, которые начинаются с s заканчиваются на $rev(s)$.
   (б) `get(s,t)` – кол-во строк в словаре, которые начинаются с s, заканчиваются на t, $|s| = |t|$.
   (в) `get(s,t)` – кол-во строк в словаре, которые начинаются с s, заканчиваются на t.
3. **LZSS.**
   Используя суффиксный массив, реализуйте сжатие `LZSS` за $\mathcal{O}(n \log n)$.
4. **Максимальные повторы. Теория.**
   Строка s называется максимальным повтором в t, если
   a) s входит в t не менее двух раз.
   b) Если r входит в t не менее двух раз, то s — не является собственной подстрокой r.

   Доказать или опровергнуть, что все максимальные повторы равны по длине.
5. **Максимальные повторы. Практика.**
   Найти все максимальные повторы за $\mathcal{O}(SA + n + ans)$.
   SA – время построения суффмассива.
   ans – количество максимальных повторов.
6. **Задачи про суффиксное дерево.**
   a) Найти количество подстрок.
   b) Найти самый длинный рефрен. Подстроку s: $count(s) \cdot |s| \to \max$.
   c) Найти за $\mathcal{O}(n)$ самую длинную подстроку, которая входит в s дважды.
   d) Найти за $\mathcal{O}(n)$ самую длинную подстроку, которая входит в s дважды, причём вхождения не пересекаются.
   e) Общая подстрока двух строк за $\mathcal{O}(|s| + |t|)$.
   f) Общая подстрока k строк за "суммарную длину всех строк".
7. **Задачи про суффиксный массив.**
   Все задачи из предыдущей серии.
8. **Бажный Укконен.**
   При подсчёте суффиксных ссылок в алгоритме Укконена маленький Петя делает спуск не по рёбрам (пройти всё ребро за шаг), а по символам (пройти один символ за шаг). Приведите пример строки, на которой полученный алгоритм будет работать дольше чем $\mathcal{O}(n)$.
9. **Амортизация в Укконене.**
   Приведите пример, когда Укконен при добавлении одного символа спустится вниз $\Omega(n)$ раз.
10. **(*) Строка по суффмассиву.**
    (а) Построить такую строку s, что её суффиксный массив совпадает с данным за $\mathcal{O}(n)$.
    (б) Минимизровать размер алфавита.

Домашнее задание

Обязательная часть

1. **(2) Разные подстроки.**
   Найти строку над алфавитом $\{0, 1\}$, в которой $\Omega(n^2)$ различных подстрок.
2. **(3) Ключевые подстроки.**
   Дан набор строк s_i.
   Для каждой s_i найдите min по длине подстроку, которая не встречается в других.
3. **(3) Уникальные суффиксы**.
   Два запроса:
   a) `addLetter(c)` – дописать в конец строки символ c.
   b) `isUnique(len)` – является ли суффикс длины len уникальной подстрокой.
4. **(3) k-я общая подстрока.**
   Найти k-ю лексикографически общую подстроку s и t за $\mathcal{O}(|s| + |t|)$.
5. **(3) Странный поиск.**
   Дан набор строк-текстов, набор строк-запросов. Для каждой строки-запроса проверить, существует ли такой текст, что и первая половина текста содержит строку целиком, и вторая половина текста содержит строку целиком. $L = \sum_i |s_i| + \sum_i |t_i|$, время работы $\mathcal{O}(L \log^k L)$, k константа.

Дополнительная часть

1. **(5) Поиск по отрезку словаря.**. Дан словарь $s_1, s_2, \ldots, s_n$. Отвечать в offline на запросы `get(t, l, r)` – сколько подстрок из $\{s_l, \ldots, s_r\}$ входят в текст t как подстроки. Время работы – линия от размера входа на полилог.
2. **(3) Дерево через автомат.** Докажите, что рёбра суффиксного дерева – суффиксные ссылки суффиксного автомата обратной строки.
3. **(5) Сумма бордеров.**
   Дана строка S. Найти за $SA + \mathcal{O}(n)$ сумму $\sum\limits_{i=1}^{n} \sum\limits_{j=i}^{n} B(S[i..j])$.
   Определение $B(S) = \max x \colon S[0..x) = S[n - x..n)$ (бордер строки).