字符串选讲 Day2

fadeawayQAQ

August 19, 2024

exKMP

exKMP

- 对于字符串 S, T 求 S 与 T 每个后缀的 LCP。
- 考虑先求 $p[i]$ 表示 $LCP(S[i:|S|], S)$ 。
- $p[i]$ 的求法是和 Manacher 类似的过程。
- 假设已经求出了 $p[1:i-1]$ ，维护最大的 $R = i + p[i] - 1$ ，$p[i]$ 的求解可以由 R 优化。
- 求和 T 的匹配是类似的过程。

最小表示法

最小表示法

- 求一个串的最小表示。
- 暴力是 $O(n^2)$ 的，考虑优化，若当前比较得知 $S[i:i+k-1]=S[j:j+k-1]$，若假设 $S[i+k]>S[j+k]$，即起始位置是 i 没有起始位置是 j 优。
  则起始位置为 $i+p$ 的循环串一定没有起始位置为 $j+p$ 的优，其中 $p\le k$。
  因此可以直接跳过起始位置为 $i+p$ 的，直接跳到 $i+k+1$，时间复杂度 $O(n)$

Lyndon

Lyndon

后缀树

伪后缀树

- 对于一个串 S，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。

后缀树

伪后缀树

- 对于一个串 S ，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。
- 每个节点对应一个子串，每个子串对应一个节点。

后缀树

伪后缀树

- 对于一个串 S，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。
- 每个节点对应一个子串，每个子串对应一个节点。
- $LCP(suf_S[i], suf_S[j])$ 即为求 LCA。

后缀树

伪后缀树

- 对于一个串 S，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。
- 每个节点对应一个子串，每个子串对应一个节点。
- $LCP(suf_S[i], suf_S[j])$ 即为求 LCA。
- 后缀排序就是后缀树的关于后缀点的 DFS 序。

后缀树

伪后缀树

- 对于一个串 S ，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。
- 每个节点对应一个子串，每个子串对应一个节点。
- $LCP(suf_S[i], suf_S[j])$ 即为求 LCA。
- 后缀排序就是后缀树的关于后缀点的 DFS 序。
- 若 x 是 y 的祖先，则 x 表示的串 P_x 是 P_y 的一个前缀。

后缀树

伪后缀树

- 对于一个串 S，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。
- 每个节点对应一个子串，每个子串对应一个节点。
- $LCP(suf_S[i], suf_S[j])$ 即为求 LCA。
- 后缀排序就是后缀树的关于后缀点的 DFS 序。
- 若 x 是 y 的祖先，则 x 表示的串 P_x 是 P_y 的一个前缀。
- P_x 的出现次数就是 x 子树里有几个后缀点。

后缀树

伪后缀树

- 对于一个串 S ，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。
- 每个节点对应一个子串，每个子串对应一个节点。
- $LCP(suf_S[i], suf_S[j])$ 即为求 LCA。
- 后缀排序就是后缀树的关于后缀点的 DFS 序。
- 若 x 是 y 的祖先，则 x 表示的串 P_x 是 P_y 的一个前缀。
- P_x 的出现次数就是 x 子树里有几个后缀点。
- P_x 在起始位置为 L 时出现当且仅当 x 是 $key[L]$ 的祖先。

后缀树

伪后缀树

- 对于一个串 S，把所有它的后缀插入到 trie 上，把每个后缀 $suf_S[i]$ 最终插入到的节点编号称为后缀点，记为 $key[i]$。
- 每个节点对应一个子串，每个子串对应一个节点。
- $LCP(suf_S[i], suf_S[j])$ 即为求 LCA。
- 后缀排序就是后缀树的关于后缀点的 DFS 序。
- 若 x 是 y 的祖先，则 x 表示的串 P_x 是 P_y 的一个前缀。
- P_x 的出现次数就是 x 子树里有几个后缀点。
- P_x 在起始位置为 L 时出现当且仅当 x 是 $key[L]$ 的祖先。
- 从 x 跳 fail 相当于删除 P_x 末尾的字符。

后缀树

伪后缀树

- 为了维护这样一棵后缀树，我们需要维护一些信息：
- $len[x]$ 表示 x 代表的串的长度。
- $fail[x]$ 表示 x 在 trie 上的父亲。
- $ch[x][c]$ 表示 cP_x 代表的节点编号（没有出现过就是 0）。

后缀树

增量构造伪后缀树

- 考虑增量法，如果已知 S 的后缀树，考虑求 cS 的后缀树。
- 我们会找到 S 代表的节点 x，然后跳 fail，即重复 $x = fail[x]$，直到 $ch[x][c]$ 不等于 0。
- 这样我们就找到了 cS 的最长的已经存在的前缀 cT 代表的节点，且此时 T 也一定是 S 的一个前缀。
- 然后我们把 $cS[1:|T|+1]...cS[1:|S|]$ 所应该产生的节点建出来即可。
- 接下来我们需要更新 $ch[x][c]$，不难发现只需要对之前跳了 fail 的每个点 y，更新 $ch[y][c]$ 即可。

后缀树

真后缀树

- 我们发现这样的后缀树点数是 $O(n^2)$ 的。
- 但是我们关注的点只有 $O(n)$ 个后缀点，也就是说我们只想保留伪后缀树的一个关于后缀点的虚树。
- 我们会压缩掉所有度为 1 的非后缀点，称为被压缩点。
- 我们称保留下来的点为关键点（即后缀点加上若干个中转点）。
- 在压缩之后，一个点表示的就是一堆串，且其中短串一定是长串的一个前缀，并且一定能取到一段连续的长度。

后缀树

真后缀树

- 此时我们称 $Z(x)$ 为 x 表示的串的集合。
- $len[x]$ 表示 x 代表的**最长**串的长度。
- $fail[x]$ 表示从 x 不断跳父亲跳到的第一个关键点。
- $ch[x][c]$ 表示 cP_x 代表的节点编号（没有出现过就是 0）。

后缀树

真后缀树的性质

- 若 x 为被压缩点，则伪后缀树上 $ch[x][c]$ 所在的点也是被压缩点。
- 证明：反证法。
- 因此被压缩点的信息一定能被关键点表示。
- 若 x 是关键点，则伪后缀树上 $ch[x][c]$ 所在的点也可能是被压缩点。
- 此时在真后缀树上我们把 $ch[x][c]$ 定为伪后缀树上 $ch[x][c]$ 表示的点被压缩到的那个关键点。
- 这样我们就保证了 $Y \in Z(x), cY \in Z(ch[x][c])$

后缀树

增量法构造真后缀树

- 考虑已经构造 S 的真后缀树，怎么构造 cS 的真后缀树。
- 像伪后缀树一样，我们要先找到一个最长的已经存在的 cT 。
- 还是像之前一样从 S 所在的点 x 不断跳 fail，直到 $ch[x][c]$ 存在。
- 但是现在我们想要找的点可能是被压缩过的，它可能在真后缀树的边上被压缩的部分里面。

后缀树

增量法构造真后缀树

- 设我们找到了点 p，使得 $q = ch[p][c]$ 非 0。
- 那么如果 $len[p] + 1 \neq len[q]$，则我们想找的 cT 就是在边上的。
- 那我们就要新建一个中转点 nq，把边拆开，让 $fail[nq] = fail[q], fail[q] = fail[np] = nq, len[nq] = len[p] + 1$。
- 其中 np 为我们新建的表示 cS 的点。

后缀树

增量法构造真后缀树

- 然后考虑更新 $ch[x][c]$ 的过程。
- p 向上跳 fail 寻找 cT 的过程中，需要把 $ch[p][c]$ 更新为 np。
- 如果新建了中转点，则其 $ch[nq][c]$ 是需要从 $ch[q][c]$ 继承的。
- 然后考虑 p 的所有祖先 pp，如果其 $ch[pp][c]=q$，则需要将 $ch[pp][c]$ 更新为 nq。

SAM

SAM

- 事实上我们这样就已经把 SAM 讲完了
- 刚才构造的真后缀树的过程实际上就是一个“前缀自动机”。
- 因为我们不断地从后往前加入 c，使得 S 变成 cS。
- 那么从前往后加字符就是 Sc，维护的 $fail[x]$ 就代表删除一段前缀（即取一段后缀）走到的点。维护的 $ch[x][c]$ 就是 Sc 代表的串所在的点。
- 也就是走 fail 就是取一段后缀，走 ch 就是末尾加一个字符。

SAM

endpos

- SAM 上每个节点 x 可以维护 endpos 集合，代表 $Z(x)$ 表示的串在原串中出现的结尾位置的集合。
- 性质：被压缩在同一个点的子串（即 $Z(x)$ 代表的所有串）的 endpos 相同，且 endpos 相同的子串都属于同一个点。
- 性质：两个点的 endpos 集合要么不交，要么包含，且包含时其中一个点代表的串是另一个点代表的串的后缀。
- 因此维护 endpos 的正确性可以保证。
- 但是 endpos 的大小之和可能是 $O(n^2)$ 的，因此一般不直接维护（必要时可以线段树合并维护）。

SAM

Code

SAM

Problem Sets

- 本质不同子串个数
- 一个串的出现次数
- $S[L:R]$ 定位
- 求最小循环串
- 求长度为 K 的字典序最小的子串
- 求 S, T 有几对子串相等
- 给定 S, T，求 LCS
- 最长的出现了至少 K 次的子串
- 求第 K 小的子串

算法回顾

KMP

算法回顾

KMP Problem 1.Rank-Kmp

- 题意：定义序列 $A[1:N]$ 和 $B[1:N]$ 相似，当且仅当对于任意 i, j，满足 $(A[i] < A[j]) = (B[i] < B[j])$ 。给定序列 X, Y，求 X 有多少个连续子序列与 Y 相似。$|X|, |Y| \le 10^6$。

算法回顾

KMP Problem 1.Rank-Kmp

- 题意：定义序列 $A[1:N]$ 和 $B[1:N]$ 相似，当且仅当对于任意 i,j，满足 $(A[i]<A[j])=(B[i]<B[j])$ 。给定序列 X,Y，求 X 有多少个连续子序列与 Y 相似。$|X|,|Y|\le 10^6$。
- 1. 考虑怎么判断相似：加一个字符的排名相等。
  2. 求排名：数据结构维护
  3. 类似于 KMP，维护 $fail[i]$ 表示对于长度为 i 的前缀适配之后最长 border（相似定义下）。

算法回顾

KMP Problem 2. 树上 KMP

- 题意：给定一棵 trie 树，求每个点到根路径表示的字符串的最长 border，节点数 $\leq 10^6$，$|\Sigma| is large$。

算法回顾

KMP Problem 2. 树上 KMP

- 题意：给定一棵 trie 树，求每个点到根路径表示的字符串的最长 border，节点数 $\leq 10^6$，$|\Sigma| islarge$。
- 做法 1：类似于 AC 自动机，维护 $fail[x]$ 和 $ch[x][c]$，表示 P_x 表示的字符串的最长 border 和 $P_x c$ 表示的字符串的最长 border。
  时间复杂度 $O(\log n)$

算法回顾

KMP Problem 2. 树上 KMP

- 题意：给定一棵 trie 树，求每个点到根路径表示的字符串的最长 border，节点数 $\leq 10^6$，$|\Sigma| is large$。
- 做法 1：类似于 AC 自动机，维护 $fail[x]$ 和 $ch[x][c]$，表示 P_x 表示的字符串的最长 border 和 $P_x c$ 表示的字符串的最长 border。
  时间复杂度 $O(\log n)$
- 做法 2：考虑当前串为 AB^k 的形式，我们要求 $AB^k c$ 的 $fail$，则若 $AB^{k-1}c$ 失配了，那么至少直到 ABc 都不可能成功匹配，所以若 $AB^{k-1}c$ 失配直接跳到 ABc 即可。
  这样每次至少减半，时间复杂度 $O(\log n)$

算法回顾

KMP Problem 3.CF1286E

- 题意：给定字符串 S 和权值数组 W。
- 定义 S 的一个子串是好的当且仅当其出现在 S 的前缀。
- 一个子串 $S[L:R]$ 的权值为 $min_{i=L}^{R} W[i]$。
- 对于每个 S 前缀求所有子串的权值之和，其中 $|S| \leq 10^5$

算法回顾

KMP Problem 3.CF1286E

- 题意：给定字符串 S 和权值数组 W。
- 定义 S 的一个子串是好的当且仅当其出现在 S 的前缀。
- 一个子串 $S[L:R]$ 的权值为 $min_{i=L}^{R} W[i]$。
- 对于每个 S 前缀求所有子串的权值之和，其中 $|S| \leq 10^5$
- 差分转化为求前缀的后缀的权值之和。
  数据结构维护以 i 结尾的前缀的所有 $border$ 以及权值。

算法回顾

AC 自动机

算法回顾

AC 自动机 Problem 4

- 题意：给定 n 个字符串 $T[1:n]$，求有多少个长度为 L 的字符串 S，使得任何一个 T_i 都不是 S 的连续子串。
- $n, |T_i| \leq 50, L \leq 1000$

算法回顾

AC 自动机 Problem 4

- 题意：给定 n 个字符串 $T[1:n]$，求有多少个长度为 L 的字符串 S，使得任何一个 T_i 都不是 S 的连续子串。
- $n, |T_i| \le 50, L \le 1000$
- AC 自动机上 dp 即可。

算法回顾

Hash

算法回顾

Manacher

算法回顾

回文自动机

算法回顾

后缀数组

算法回顾

后缀数组 Problem Sets

- 求 $S[L:R]$ 出现次数。
- 求本质不同子串个数。
- 求字符串最小循环表示
- 求最长的出现了至少 K 次的子串

算法回顾

后缀数组 Problem Sets

- 求 $S[L:R]$ 出现次数。
- 求本质不同子串个数。
- 求字符串最小循环表示
- 求最长的出现了至少 K 次的子串
- 求最长的出现了至少 K 次的子串，要求不重叠。
- 求 S, T 的 LCS（最长公共子串）
- 求第 k 小子串。

算法回顾

后缀数组 Problem 1.BZOJ4310

- 题意：给定字符串 S，将他分成不超过 m 个连续子串，使得分割后所有串的子串中字典序最大的尽量小。

算法回顾

后缀数组 Problem 2.BZOJ4556

- 题意：给定字符串 S，Q 次询问，每次给出 a, b, c, d，询问 $S[a:b]$ 所有子串和 $S[c:d]$ 的 LCP 的最大值。
- $|S| \leq 10^5$

算法回顾

后缀自动机 Problem 1

- 题意：给定 S, 设 $f(i)$ 为长度为 i 的子串中出现次数的最大值。求 $f(1)...f(|S|)$
- $|S| \leq 3 * 10^5$ ，要求线性。

算法回顾

后缀自动机 Problem 2

- 题意：维护一个串 S，支持往后加 a，以及多次给出 T，询问 T 在 S 中的出现次数。$Q \leq 10^5$。

算法回顾

后缀自动机 Problem 3

- 题意：给定 n 个字符串，求有多少个不同的字符串是至少 K 个给定串的子串。$\sum |S_i| \leq 10^5, n, k \leq 100$

算法回顾

后缀自动机 Problem 4.BZOJ4556

- 题意：给定字符串 S，Q 次询问，每次给出 a, b, c, d，询问 $S[a:b]$ 所有子串和 $S[c:d]$ 的 LCP 的最大值。
- $|S| \leq 10^5$

一些和字符串有一点关系的题

Problem 1.CF2004F

- 题意：给定一个长度为 n 的序列 a_i，有两种操作：
  1. 删除相邻的数并替换为两个数的和。
  2. 把一个数替换为两个数，要求两个数的和等于原数。
  定义 $f(l,r)$ 为操作 $a_l \ldots a_r$ 需要的最小操作数，求 $\sum_{i=1}^{n}\sum_{j=i}^{n} f(i,j)$。

一些和字符串有一点关系的题

Problem 1.CF2004G

思路：每隔 k 位求前缀和后缀矩阵积。