字符串 & tricks

长郡中学 周书予

2019 年 9 月 27 日

【数据删除】

由于【数据删除】的原因，我们先讲 tricks。

一些约定

- 如非特殊说明，所有字符串的下标均从 1 开始编号。
- σ 表示字符集大小，一般默认为 26。
- $S+T$ 表示字符串 S 与字符串 T 的拼接。
- $|S|$ 表示字符串 S 的长度。
- S_i 表示字符串 S 的第 i 个字符。
- $S_{i\ldots j}$ 表示字符串 S 第 i 个字符到第 j 个字符组成的子串。
- pre_i 表示字符串 S 长度为 i 的前缀，即 $S_{1\ldots i}$。
- suf_i 表示字符串 S 长度为 i 的后缀，即 $S_{|S|-i+1\ldots|S|}$。
- 定义一个串存在长度为 i 的 border 当且仅当 $pre_i = suf_i$。
- $lcp(i,j)$ 表示 suf_i 与 suf_j 的最长公共前缀。
- $lcs(i,j)$ 表示 pre_i 与 pre_j 的最长公共后缀。

最小循环表示法

给一个字符串 $S(|S| \leq 10^7)$，求一个 i 使 $S_{i...|S|} + S_{1...i-1}$ 的字典序最小。

最小循环表示法

给一个字符串 $S(|S| \leq 10^7)$，求一个 i 使 $S_{i...|S|} + S_{1...i-1}$ 的字典序最小。

将 S 倍长，维护两个指针 i, j 表示最小循环表示的起始位置，暴力求出 $lcp(i, j)$，设其为 k，则比较 S_{i+k} 和 S_{j+k}，不失一般性地假设 $S_{i+k} < S_{j+k}$，则 $[j, j+k]$ 范围内的所有下标均不可能成为最小循环表示的起始位置，直接令 $j \leftarrow j+k+1$ 后继续比较，直至 $[1, n]$ 内仅剩下一个起始位置。

最小循环表示法

给一个字符串 $S(|S| \leq 10^7)$，求一个 i 使 $S_{i...|S|} + S_{1...i-1}$ 的字典序最小。

将 S 倍长，维护两个指针 i, j 表示最小循环表示的起始位置，暴力求出 $lcp(i, j)$，设其为 k，则比较 S_{i+k} 和 S_{j+k}，不失一般性地假设 $S_{i+k} < S_{j+k}$，则 $[j, j+k]$ 范围内的所有下标均不可能成为最小循环表示的起始位置，直接令 $j \leftarrow j+k+1$ 后继续比较，直至 $[1, n]$ 内仅剩下一个起始位置。

由于指针 i, j 始终是递增的，因而该算法的时间复杂度为 $O(|S|)$。

manacher

manacher 算法可以对串 S 的每个位置 i，求出以其为中心的最长奇回文串长度。若需要求所有回文串，则可以在串 S 的任意两个相邻字符中间插入一种新的字符。

manacher

manacher 算法可以对串 S 的每个位置 i，求出以其为中心的最长奇回文串长度。若需要求所有回文串，则可以在串 S 的任意两个相邻字符中间插入一种新的字符。

记 len_i 表示以位置 i 为中心的最长奇回文串半径，考虑从左往右依次求解 len_i。记录前缀中 $i + len_i$ 的最大值 mx 及其对应的位置 pos，当需要计算 len_j 时，根据回文的对称性可以利用到之前已得到的信息，即 $len_j \geq \min(len_{2\times pos-j}, mx - j)$，这样每次暴力扩展都必然导致 mx 增大，从而使复杂度均摊 $O(|S|)$。

ex-kmp

ex-kmp 又称 Z-algorithm，可以对串 S 求出所有 $lcp(1,i), i \in [2,|S|]$，即原串与所有后缀的最长公共前缀。

ex-kmp

ex-kmp 又称 Z-algorithm，可以对串 S 求出所有 $lcp(1,i), i \in [2,|S|]$，即原串与所有后缀的最长公共前缀。

为什么先讲 ex-kmp 后讲 kmp？是因为 ex-kmp 跟 kmp 没什么太大关系，反倒是和 manacher 很相似。

ex-kmp

ex-kmp 又称 Z-algorithm，可以对串 S 求出所有 $lcp(1,i), i \in [2,|S|]$，即原串与所有后缀的最长公共前缀。

为什么先讲 ex-kmp 后讲 kmp？是因为 ex-kmp 跟 kmp 没什么太大关系，反倒是和 manacher 很相似。

类似 manacher，记 len_i 表示 $lcp(1,i)$，同时记录 $i+len_i$ 的最大值 mx 及其对应位置 pos，可以发现 $len_j \geq \min(len_{j-pos+1}, mx-j)$，其余部分与 manacher 几乎完全相同。

hash

把一个字符串映射到一个数。通过构造 hash 函数以尽量减少冲突，理想情况是使映射趋于完全随机。

hash

把一个字符串映射到一个数。通过构造 hash 函数以尽量减少冲突，理想情况是使映射趋于完全随机。

常见的构造函数形如：

$$hash(S) = \sum_{i=1}^{|S|} S_i \times Base^{|S|-i} \mod p$$

p 可以取一个大质数，也可以取 2^{32} 或 2^{64}，不过值得注意的是确定存在并且已经构造出了能使 $p = 2^{64}$ 产生 hash 冲突的字符串。

hash 表

既然讲到了 hash 就不得不提一下 hash 表。

把一个（不知道什么东西）近似随机地映射到一个比较小的数，然后用数组存下来。

hash 表

既然讲到了 hash 就不得不提一下 hash 表。

把一个（不知道什么东西）近似随机地映射到一个比较小的数，然后用数组存下来。

可能会存在冲突，那么就可以在数组上挂链，或者直接暴力往后找直到找到一个未被占用的位置。

hash 表

既然讲到了 hash 就不得不提一下 hash 表。

把一个（不知道什么东西）近似随机地映射到一个比较小的数，然后用数组存下来。

可能会存在冲突，那么就可以在数组上挂链，或者直接暴力往后找直到找到一个未被占用的位置。

如果随机构造得足够优秀的话可以做到期望 $O(1)$ 的插入查询复杂度。

kmp

kmp 的本质是对每一个 i 求出最大的 $j < i$ 满足 $S_{i-j+1\ldots i} = pre_j$，记之为 nxt_i。

kmp

kmp 的本质是对每一个 i 求出最大的 $j < i$ 满足 $S_{i-j+1\ldots i} = pre_j$，记之为 nxt_i。

求 nxt_i 时，从 $i-1$ 开始不断跳 nxt，直到找到一个下一位与 S_i 相同的位置或跳到了 0 为止。（具体见板书）

kmp

kmp 的本质是对每一个 i 求出最大的 $j < i$ 满足 $S_{i-j+1\ldots i} = pre_j$，记之为 nxt_i。

求 nxt_i 时，从 $i-1$ 开始不断跳 nxt，直到找到一个下一位与 S_i 相同的位置或跳到了 0 为止。（具体见板书）

注意到 $nxt_i \leq nxt_{i-1} + 1$，因此 nxt 指针的总移动距离是 $O(n)$ 的，这也说明 kmp 算法的时间复杂度是**均摊**线性的。

kmp 自动机

在 kmp 求 nxt 的基础上，额外求出 $trans_{i,j}$ 表示从 i 位置开始往后匹配一个 j 字符后会转移到什么状态，可以实现真正严格 $O(1)$ 而非均摊 $O(1)$ 的转移。

Trie

又称字典树，结构是除根节点以外每个点上有一个字符（一种说法是每条边上有一条字符），每个节点所代表的字符串就是从根节点出发到该节点路径上所有字符顺次连接形成的串。

Trie

又称字典树，结构是除根节点以外每个点上有一个字符（一种说法是每条边上有一条字符），每个节点所代表的字符串就是从根节点出发到该节点路径上所有字符顺次连接形成的串。

两个串的最长公共前缀长度即为对应的两点在 Trie 树上 LCA 的深度。

Aho-Corasick automaton

本质是 Trie 上实现 kmp 自动机。

Aho-Corasick automaton

本质是 Trie 上实现 kmp 自动机。

按照深度由浅到深做。对每个节点求出 $fail_i$ 表示这个点对应的串在 Trie 树中存在的最长严格后缀，构造方式也与 kmp 类似，通过不断跳 $fail$ 以找到一条与当前字符相同的出边。(具体见板书)

Aho-Corasick automaton

本质是 Trie 上实现 kmp 自动机。

按照深度由浅到深做。对每个节点求出 $fail_i$ 表示这个点对应的串在 Trie 树中存在的最长严格后缀，构造方式也与 kmp 类似，通过不断跳 $fail$ 以找到一条与当前字符相同的出边。(具体见板书)

一般的写法是直接建出自动机，在构建时只需要从其 $fail_i$ 转移而来即可。

Aho-Corasick automaton

本质是 Trie 上实现 kmp 自动机。

按照深度由浅到深做。对每个节点求出 $fail_i$ 表示这个点对应的串在 Trie 树中存在的最长严格后缀，构造方式也与 kmp 类似，通过不断跳 $fail$ 以找到一条与当前字符相同的出边。(具体见板书)

一般的写法是直接建出自动机，在构建时只需要从其 $fail_i$ 转移而来即可。

同时，根据 $i \to fail_i$ 可以建出一个树形结构，这棵树被称为 $fail$ 树，它具有一些优美的性质可以拿来出题。

snoi2019 字符串

description

给一个长度为 n 的字符串 S，记 S'_i 表示 S 删去第 i 个字符后得到的字符串，输出 $(S'_1, 1)...(S'_n, n)$ 的字典序排序结果。

constriction

$1 \leq n \leq 10^6$.

snoi2019 字符串

description

给一个长度为 n 的字符串 S，记 S'_i 表示 S 删去第 i 个字符后得到的字符串，输出 $(S'_1, 1)...(S'_n, n)$ 的字典序排序结果。

constriction

$1 \leq n \leq 10^6$.

solution

考虑比较 S'_i 和 S'_j 的字典序大小，发现只需要用到 $lcp(\min(i,j), \min(i,j)+1)$，即相邻两个后缀的 lcp。这个可以从后往前 $O(n)$ 预处理出来。

thupc2018A 绿绿与串串

description

对于字符串 S，定义运算 $f(S)$ 为将 S 的前 $|S|-1$ 个字符倒序接在 S 后面形成的长为 $2|S|-1$ 的新字符串。
现给出字符串 T，询问有哪些串长度不超过 $|T|$ 的串 S 经过若干次 f 运算后得到的串包含 T 作为前缀。

constriction

$1 \leq |T| \leq 5 \times 10^6$.

thupc2018A 绿绿与串串

description

对于字符串 S，定义运算 $f(S)$ 为将 S 的前 $|S|-1$ 个字符倒序接在 S 后面形成的长为 $2|S|-1$ 的新字符串。
现给出字符串 T，询问有哪些串长度不超过 $|T|$ 的串 S 经过若干次 f 运算后得到的串包含 T 作为前缀。

constriction

$1 \leq |T| \leq 5 \times 10^6$.

solution

合法的串 S 一定是 T 的前缀。
先用 manacher 求出每个位置的回文半径，从后往前处理，一个位置合法当且仅当其回文半径右侧达到 $|T|$，或者左侧达到 1 且右侧位置合法。

ctsc2014 企鹅 QQ

description

给 n 个长度相同且互不相同的串，询问有多少对串仅在一个位置上不同。

constriction

$1 \leq n \leq 30000, 1 \leq |S_i| \leq 200.$

ctsc2014 企鹅 QQ

description

给 n 个长度相同且互不相同的串，询问有多少对串仅在一个位置上不同。

constriction

$1 \leq n \leq 30000, 1 \leq |S_i| \leq 200.$

solution

直接枚举哪一位不同，hash 前后缀即可。

cqoi2014 通配符匹配

description

给出一个带通配符的字符串 S，通配符分两种，一种可以匹配恰好一个字符，另一种可以匹配任意个（包括 0 个）字符。再给出 n 个串 T_i，询问 T_i 能否与 S 匹配。

constriction

$1 \le n \le 100, 1 \le |S|, |T_i| \le 10^5, 0 \le$ 通配符个数 ≤ 10.

cqoi2014 通配符匹配

description

给出一个带通配符的字符串 S，通配符分两种，一种可以匹配恰好一个字符，另一种可以匹配任意个（包括 0 个）字符。再给出 n 个串 T_i，询问 T_i 能否与 S 匹配。

constriction

$1 \le n \le 100, 1 \le |S|, |T_i| \le 10^5, 0 \le$ 通配符个数 ≤ 10.

solution

设 $f_{i,j}$ 表示前 i 个通配符匹配 T 的前 j 个字符是否可行，用 hash 判断字符串相等即可。转移“匹配任意个字符”的通配符时需要做一次前缀和。

hnoi2008 GT 考试

description

给一个数字串 S，求有多少长度为 n 的数字串 T 不包含 S 作为子串。

constriction

$1 \leq |S| \leq 100, 1 \leq n \leq 10^9$.

hnoi2008 GT 考试

description

给一个数字串 S，求有多少长度为 n 的数字串 T 不包含 S 作为子串。

constriction

$1 \leq |S| \leq 100, 1 \leq n \leq 10^9$.

solution

对 S 建出 kmp 自动机，把匹配到串 S 的每个位置视作一个状态，转移是在串的末尾加一个字符，相当于是在 kmp 自动机上走一条出边，矩阵快速幂转移即可。

noi2011 阿狸的打字机

description

给你一棵 n 个节点的 Trie 树，q 次询问 Trie 树上 x 节点代表的字符串在 y 节点代表的字符串中出现了多少次。

constriction

$1 \leq n, q \leq 10^6$.

noi2011 阿狸的打字机

description

给你一棵 n 个节点的 Trie 树，q 次询问 Trie 树上 x 节点代表的字符串在 y 节点代表的字符串中出现了多少次。

constriction

$1 \leq n, q \leq 10^6$.

solution

问题等价于问 y 节点代表的字符串有多少个前缀包含 x 作为后缀，也就是问 Trie 树上根节点到 y 路径上的所有点中有多少点在 $fail$ 树上 x 的子树里，树状数组维护即可。

搜索

应该算是大家最早接触的算法？

搜索

应该算是大家最早接触的算法？

深度优先搜索/广度优先搜索？

搜索

应该算是大家最早接触的算法？

深度优先搜索/广度优先搜索？

状态的含义/表示方法？状态压缩？`map<vector<vector<int>>,int>`？

搜索

应该算是大家最早接触的算法？

深度优先搜索/广度优先搜索？

状态的含义/表示方法？状态压缩？`map<vector<vector<int>>,int>`？

转移量？注意搜索的复杂度是与转移总量有关而非与状态总数有关。

搜索

应该算是大家最早接触的算法？

深度优先搜索/广度优先搜索？

状态的含义/表示方法？状态压缩？`map<vector<vector<int>>,int>`？

转移量？注意搜索的复杂度是与转移总量有关而非与状态总数有关。

剪枝是个比较玄学的东西，有的时候可以据此分析得到正确的复杂度。

搜索

应该算是大家最早接触的算法？

深度优先搜索/广度优先搜索？

状态的含义/表示方法？状态压缩？`map<vector<vector<int>>,int>`？

转移量？注意搜索的复杂度是与转移总量有关而非与状态总数有关。

剪枝是个比较玄学的东西，有的时候可以据此分析得到正确的复杂度。

双向搜索：从两边开始搜以减少状态数（通常是指数上的折半），深搜广搜都能用。

hncpc2019E Numbers

description

给一个数字串 S，求将其划分成 $[0, 99]$ 不含前导零且互不相同的数的方案数。

constriction

$1 \leq |S| \leq 50$.

hncpc2019E Numbers

description

给一个数字串 S，求将其划分成 $[0,99]$ 不含前导零且互不相同的数的方案数。

constriction

$1 \leq |S| \leq 50$.

solution

直接搜。

个位数（包括 0）只有 10 个，确定了个位数的位置后剩下位置的划分也就确定了。

设数字 i 出现了 a_i 次，搜索的总状态数不超过 $\prod_{i=0}^{9}(a_i+1) \leq (\frac{\sum_{i=0}^{9}(a_i+1)}{10})^{10} = 6^{10}$。

loj6043 蛐蛐国的修墙方案

description

给一个 n 阶排列 p，需要构造一个长度为 n 的合法括号序列，满足：构造一张 n 个点的图，当且仅当第 i 个位置是左括号时，点 i 向点 p_i 连边，要求最终形成的图中每个点的度数均为 1。

constriction

$1 \leq n \leq 100, n$ 是偶数。

loj6043 蛐蛐国的修墙方案

description

给一个 n 阶排列 p，需要构造一个长度为 n 的合法括号序列，满足：构造一张 n 个点的图，当且仅当第 i 个位置是左括号时，点 i 向点 p_i 连边，要求最终形成的图中每个点的度数均为 1。

constriction

$1 \leq n \leq 100, n$ 是偶数。

solution

考虑 i 向 p_i 连边，这样的图一定是由若干个环构成，且每个点的度数为 2。因此为了有解，每个环的大小都应为偶数，且选取方案有且仅有两种（考虑一条边选或不选，其余的边随之确定）。

这样就可以做到 $O(2^{\frac{n}{2}})$ 的搜索复杂度。考虑到当环大小为 2 时，在左边放左括号一定不会更劣，所以就只需要考虑所有大小大于等于 4 的环即可，复杂度 $O(2^{\frac{n}{4}})$。

bzoj4722 由乃

description

维护一个长度为 n 的序列 $\{a_i\}$，支持 m 次如下两种操作：

- 修改：给出 l, r，将 $[l, r]$ 内的所有 a_i 替换成 $a_i^3 \bmod v$。
- 查询：给出 l, r，询问是否能从 $\{a_l + 1, a_{l+1} + 1, ..., a_r + 1\}$ 中选出两个不同的子集元素和相等。

constriction

$1 \leq n, m \leq 10^5, 1 \leq v \leq 10^3, 0 \leq a_i < v.$

bzoj4722 由乃

description

维护一个长度为 n 的序列 $\{a_i\}$，支持 m 次如下两种操作：

- 修改：给出 l, r，将 $[l, r]$ 内的所有 a_i 替换成 $a_i^3 \bmod v$。
- 查询：给出 l, r，询问是否能从 $\{a_l + 1, a_{l+1} + 1, ..., a_r + 1\}$ 中选出两个不同的子集元素和相等。

constriction

$1 \le n, m \le 10^5, 1 \le v \le 10^3, 0 \le a_i < v.$

solution

一个长度为 len 的区间内子集元素和的值域是 $[0, len \times v]$，共有 2^{len} 个子集，根据鸽巢原理，当 $2^{len} > len(v-1)+1$ 即 $len \ge 14$ 时，答案一定为 `Yes`。否则直接 $O(3^{\frac{len}{2}})$ 双搜。至于修改操作，只需要对每个位置记一下被修改了多少次，维护一个 $x \to x^3 \bmod v$ 的倍增数组即可。

cf1105E Helping Hiasat

description

给一张 n 点 m 条边的图，点有点权，求这张图的最大权独立集。

constriction

$1 \leq n \leq 50, 0 \leq m \leq \frac{n(n-1)}{2}$.

cf1105E Helping Hiasat

description

给一张 n 点 m 条边的图，点有点权，求这张图的最大权独立集。

constriction

$1 \leq n \leq 50, 0 \leq m \leq \frac{n(n-1)}{2}$.

solution

将点集划分成两部分。对于前一部分，求出每个子集内的最大权独立集，接着直接搜索后一部分点集的选取，并状压表示出选出的点集在前一部分的相邻点集。时间复杂度为 $O(2^{\frac{n}{2}})$。

loj6487 基础 FFT 练习题

description

给两个长度为 n **的数组** $\{a_i\},\{b_i\}$，**求**

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\lfloor\sqrt{|a_i-b_j|}\rfloor$$

constriction

$1\le n\le 10^5, 1\le \sum_{i=1}^{n} a_i, \sum_{i=1}^{n} b_i\le 10^6$, **时间限制** $70ms$.

loj6487 基础 FFT 练习题

description

给两个长度为 n 的数组 $\{a_i\},\{b_i\}$，求

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\lfloor\sqrt{|a_i-b_j|}\rfloor$$

constriction

$1\le n\le 10^5, 1\le \sum_{i=1}^{n} a_i, \sum_{i=1}^{n} b_i\le 10^6$, 时间限制 $70ms$.

solution

由于 $\sum_{i=1}^{n} a_i, \sum_{i=1}^{n} b_i\le 10^6$，因此数值不同的 a_i, b_i 的数目是 $\sqrt{10^6}$ 级别的。
合并值相同的 a_i, b_i 后暴力即可。需要预处理开根的结果以减小常数。

随机化

有些问题并不能够在我们期望的时间复杂度内得到正确的解，不过可能存在一些做法可以得到准确度较高的解，或者以较高的概率得到正确的解。
~~我实在不知道这里该写什么了所以还是看几道题吧。~~

pa2013 Filary

description

有 n 个数 $a_1, a_2, ..., a_n$，你需要确定两个数 $k, b(k \geq 2, 0 \leq b < k)$ 使得满足 $a_i \equiv b \bmod k$ 的 i 尽量多。

constriction

$1 \leq n \leq 10^5, 1 \leq a_i \leq 10^7$.

pa2013 Filary

description

有 n 个数 $a_1, a_2, ..., a_n$，你需要确定两个数 $k, b(k \geq 2, 0 \leq b < k)$ 使得满足 $a_i \equiv b \bmod k$ 的 i 尽量多。

constriction

$1 \leq n \leq 10^5, 1 \leq a_i \leq 10^7$.

solution

随机选一个数 a_x，计算强制其被选的方案。把所有其他数与之作差，相当于是要找尽量多的数它们的 $\gcd$ 不为 1，分解质因数后，统计每个质因子有多少个倍数即可。
可以发现答案的 k 至少为 $\lceil \frac{n}{2} \rceil$（在 $k = 2$ 时即可取到），因此随机选一个数能得到正确答案的概率是 $\frac{1}{2}$，随机 k 次后正确率即可达到 $1 - \frac{1}{2^k}$。

bzoj3569 DZY Loves Chinese II

description

给一张 n 点 m 条边的无向连通图，q 次询问，每次删除 k 条边，询问图是否仍然连通。强制在线。

constriction

$1 \leq n, q \leq 10^5, 1 \leq m \leq 5 \times 10^5, 1 \leq k \leq 15.$

bzoj3569 DZY Loves Chinese II

description

给一张 n 点 m 条边的无向连通图，q 次询问，每次删除 k 条边，询问图是否仍然连通。强制在线。

constriction

$1 \leq n, q \leq 10^5, 1 \leq m \leq 5 \times 10^5, 1 \leq k \leq 15.$

solution

随便找棵生成树。对每条非树边随机一个 $[0, 2^\omega)$ 的权值，再令每条树边的权值为所有覆盖它的非树边的权值异或和。这样图不连通当且仅当删掉的边权集合线性相关。具体证明可以参考这篇博客

求删除的所有边中是否存在一个边集的权值异或和为 0 即可，使用线性基解决，复杂度 $O(n + m + qk\omega)$。这个做法的正确率大概是 $(1 - \frac{1}{2^\omega})^{2^k}$。

谢谢大家！