NOI模拟赛Solution

samjia2000

June 12, 2018

祝大家NOI取得理想的成绩。

1 一道很好玩的计算几何题

1.1 Description

给出n，要求构造一个序列$h_0, h_1, \cdots, h_{n-1}, h_n$，使得对于$i(0 \le i < n)$，$h_{i+1} = h_i$与$h_{i+1} = h_i - 1$必有一个成立，并且$h_n = 0$，要求最大化$\{(i, h_i) | 0 \le i \le n\}$的上凸壳的点的个数。

$1 \le n \le 1000000$

1.2 Solution

1.2.1 $1 \le n \le 20$

暴力枚举h，然后求上凸壳，时间复杂度$O(2^n n \ log \ n)$。

1.2.2 $1 \le n \le 100$

假设最终的关键点序列是$x_1, x_2, \cdots, x_k$，为了方便，假设n也是关键点（即$x_k = n$）。

对于$i(1 \le i < k)$，定义$\vec{v_i} = (x_{i+1} - x_i, h_{x_i} - h_{x_{i+1}})$（为了方便叙述，笔者有意调整了$h$和$x$的顺序，使得所有向量都在第一象限内或$x$轴正半轴上）。

对于$\{\vec{v_i}\}$有如下性质：

- 当i递增的时候，$\vec{v_i}$与x轴的夹角也在不断的递增。
- 除了最后一个向量之外，每个向量的x坐标必定大于它的y坐标。
- 不存在两个向量是共线的。

当$1 \le n \le 100$的时候，可行的向量不会很多，只需要暴力枚举每个向量是否取就可以了，注意判断向量$(1, 0)$与$(1, 1)$是否取。

1.2.3 $1 \le n \le 1000$

当然，也可以直接做背包求出最多用多少不同的向量，时间复杂度$O(n^2)$。

1.2.4 $1 \le n \le 100000$

这档是给正解写TLE的选手的，~~不过奶一口，应该不存在这样的选手吧~~。

1.2.5 $1 \le n \le 1000000$

如果确定了最后的向量是什么，就可以直接通过极角排序来还原h序列。

所以问题就变成了，要选择尽量多的不共线向量，这些向量(x, y)必须满足$0 \le y \le x$，并使这些向量的x坐标加起来恰好为n。

不妨贪心的想，最终一定会选择某个$(p, 0)$，假设一开始是$(1, 0)$，实际上上面条件的**恰好为**n可以改写成**不超过**n，不够的部分可以加在向量$(1, 0)$上。

更进一步的，实际上，当$gcd(x, y) > 1$的时候，向量(x, y)是一定不会被选择的，设$d = gcd(x, y)$，选择$(\frac{x}{d}, \frac{y}{d})$比选择$(x, y)$更优。

所以，选择的向量(x, y)除了$(1, 1)$以及$(p, 0)$外，必定满足：

- $0 < y < x$
- $gcd(x, y) = 1$

可以发现，y的作用仅在于限制x，而不会对x**坐标加起来不超过**n产生影响，所以，就可以得到一个十分简单的算法：

先把$(1, 0)$加进答案里，然后从小到大枚举x，枚举$y(0 < y < x)$，如果$gcd(x, y) = 1$，那么就把(x, y)放进答案向量里面，当答案向量里面不能加入新的向量的时候就停止这个过程，最后特判能否加入$(1, 1)$，然后如果x的和小于n，就把不够的部分加到$(1, 0)$上面，极角排序之后还原出h数组即可。

时间复杂度$O(n\ log\ n)$。

2 糖果

2.1 Description

给出函数$f(i)$,$f(i)=\sum_{x=1}^{n} c_x \times f(i-x)$，$g(i)=\sum_{x=1}^{n} g(i-x)\times f(x)$，求$g_m$。

$1 \le n \le 50000, 1 \le m \le 10^9$

2.2 Solution

2.2.1 $1 \le n, m \le 1000$

直接照着式子$O(n^2)$计算即可。

2.2.2 $1 \le n \le 100$

稍稍了解多项式的同学可以猜到$g(i)$也是有一个类似$f(i)$的递推式的，高斯消元解出来之后就可以矩阵快速幂求答案了。

时间复杂度$O(n^3\ log\ m)$

2.2.3 $1 \le m \le 50000$

首先，设$f(i)$的生成函数是$F(x)$，$g(i)$的生成函数是$G(x)$。

那么，$F(x)=\frac{A(x)}{M(x)}$，其中$M(x)=1-\sum_{i=1}^{n} c_i \cdot x^i$，$[x^i]A(x)=a_i-\sum_{j=1}^{i-1} a_j \times c_{i-j}$。

对于$G(x)$有$G(x)=\frac{1}{1-F(x)}=\frac{M(x)}{M(x)-A(x)}$。

由于$m \le 50000$，求这个的第m项可以直接求逆得到。

时间复杂度$O((n+m)\ log\ n)$

2.2.4 $1 \le n \le 1000$

可以发现，$G(x)$的分母是一个首一多项式，这说明可以用求一般常系数其次线性递推的方法求$G(x)$的某一项。

$G(x)$的分母就是其特征多项式。

设$P(x)=x^n \cdot (M(x^{-1})-A(x^{-1}))$，快速幂求出$x^m$对$P(x)$取模的结果。

不会写/不想写/太强了不屑于写多项式取模/多项式求逆的同学，可以直接$O(n^2)$暴力卷积/取模/求逆。

时间复杂度$O(n^2\ log\ m)$。

2.2.5 $1 \le n \le 50000$

将上面的多项式操作都优化掉就可以通过本题了。
时间复杂度$O(n\ log\ n\ log\ m)$。
~~这不就是传说中的多项式模板题吗~~

3 一道很好玩的字符串题

3.1 Description

有一个长度为n的字符串S，有q组询问，每组询问给出l,r，要求求一个最大的x满足$S[l..l+x-1]$与$S[r-x+1..r]$相同，且要求$x \le r-l$。

$1 \le n,q \le 200000$

3.2 Solution

3.2.1 $1 \le n,q \le 1000$

这档是给常数较大的$O(n^2)$做法或$O(n^2\ log\ n)$做法的。

3.2.2 $1 \le n \le 10000$

对于$1 \le l \le n$，处理出对应的所有r的答案，直接做kmp就可以了。

时间复杂度$O(n^2)$。

3.2.3 S中每个字符有0.5的概率是a,有0.5的概率是b

对S建立后缀自动机，那么$parent$树树高期望是$log\ \ n$的，对于每个询问(l,r)，沿着$S[1..r]$到根的路径上去，对于每个路径上的节点u，查询u的$Right$集中，小于等于$min(l+len[u]-1,r-1)$的最大值，主席树可以轻松解决。

时间复杂度$O(n\ log^2\ n)$。

~~可能会存在花式乱水~~

3.2.4 $1 \le n,q \le 50000$

设后缀自动机中$Right$集为$\{r\}$的节点是key_r。

先对$parent$树做树链剖分。

考虑询问(l,r)，假设其最优的位置是$w=l+x-1$，设t是key_w与key_r的lca，w与r的关系可以分两种情况讨论：

- t到key_r的路径上的第一条边是虚边。
  这种情况下，从key_r出发，每次跳到所在重链顶点的父亲，这样经过的所有点都是这种情况下可能的lca，设跳到的点是u，查询就只需要在u的$Right$集中，找小于等于$min(l+len[u]-1,r-1)$的最大值，主席树可以轻松解决，时间复杂度$O(n\ log^2\ n)$。

- t到key_w的路径上的第一条边是虚边。
  类似的，从key_w出发，每次跳到所在重链顶点的父亲，这样经过的所有点都是这种情况下可能的lca，设跳到的点是u，那么就需要对右端点在u的$Right$集中的询问更新答案，具体来说需要满足$w-len_u+1 \leq l$且$w < r$，这样需要用二维数据结构维护，时间复杂度$O(n\ log^3\ n)$。

尽管上面两种情况有交集，但是也不影响答案的正确性。
时间复杂度$O(n\ log^3\ n)$。

3.2.5 $1 \leq n, q \leq 200000$

在$1 \leq n, q \leq 50000$的部分分中，复杂度较劣的部分在于处理**"t到key_w的路径上的第一条边是虚边"**的情况，现在尝试优化这一部分。

考虑从大到小枚举w，沿着key_w的树链往上跳到节点u的时候，当某个询问(l, r)满足$w - len_u + 1 \leq l$且$w < r$，在前面的做法中，我们会更新这个询问的答案，但实际上，当询问(l, r)第一次满足这个条件的时候，现在的w就是这种情况下这个询问最优的答案，因为当w更大的时候没有找到满足的，当w更小的时候显然更劣，所以，当一个询问第一次被满足的时候就已经找到了最优解。

那么，对于这种情况，我们就可以得到一个优秀的$O(n\ log^2\ n)$的做法：

从大到小枚举w，此时，所有$r > w$的还没有找到最优解的询问(l, r)都已经挂在了节点key_r上，用线段树维护dfs序区间内l的最大值，然后对于w，从key_w不断往上跳，对于跳到的每个节点u，在线段树上查询这个子树区间的最大值，如果这个最大值大于等于$w - len_u + 1$，那么说明这个询问找到了最优解，更新答案，并把这个询问从树上面删掉，处理完w之后，把所有$r = w$的询问都插入到树上。

时间复杂度$O(n\ log^2\ n)$。

~~小清新字符串题~~

3.2.6 一个十分优秀的$O(n\sqrt{n}\log\ n)$算法

这是一个完全不同的做法，虽然复杂度很大但是实际效率很优秀，常数很小，代码也比较简洁~~（辣鸡std，辣鸡出题人）~~，这里简单介绍一下这个算法的思路。

将S分成$\sqrt{n}$块，对于每一块$[L, R)$，处理出l在$[L, R)$内的询问的答案。

对于一个这样的询问(l, r)，如果最终答案小于等于$R - l$，那么可以直接暴力，否则，最优的答案会被分成两部分，一部分是$S[l, R-1]$，一部分是$S[R, l+x-1]$。

对于每个位置i，用扩展kmp处理出pre_i表示一个最大的x，使得$S[R-x,R-1]$与$S[i-x+1,i]$相同，以及suf_i表示一个最大的x，使得$S[R,R+x-1]$与$S[i,i+x-1]$相同。

然后，从大到小枚举$S[l,R-1]$的长度x，然后，对于每个$pre_i=x$的位置i，在数据结构上$i+suf_{i+1}$的位置插入i，对于一个询问，由于$S[l,R-1]$的长度是已经确定了的，那么就只需要找到一个最小的i满足$pre_i \geq x$且$i+suf_{i+1} \geq r$，这个直接在刚才的数据结构上面查询即可，由于是查询后缀最值，所以可以用相对来说常数较小的树状数组维护。

时间复杂度$O(n\sqrt{n}\,log\,n)$。