IOI2016试题讨论

吴作凡(2952643618@qq.com)

安徽师范大学附属中学

2017年2月4日

Tetris Statement

相信大家都玩过俄罗斯方块，现在让你在3列的网格中进行游戏，有三种不同的方块如下：

Type	Figure
1	
2	
3	

共会有不超过10000个方块，你可以旋转方块并将其从上面任意一个位置扔下，如果某一行的三列被填满就会消失，你要保证时刻不会有方块超过第四行。

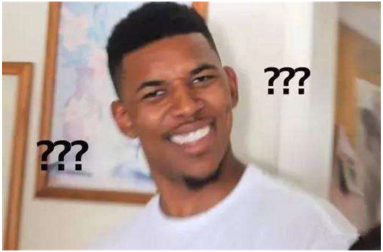
Solution

首先可以不考虑第一种方块，通过构(xia)造(wan)可以使得后两种方块所构成的情况很少，然后分情况讨论即可。

Reverse Statement

给你一个长度不超过100000的数组，将其翻转。

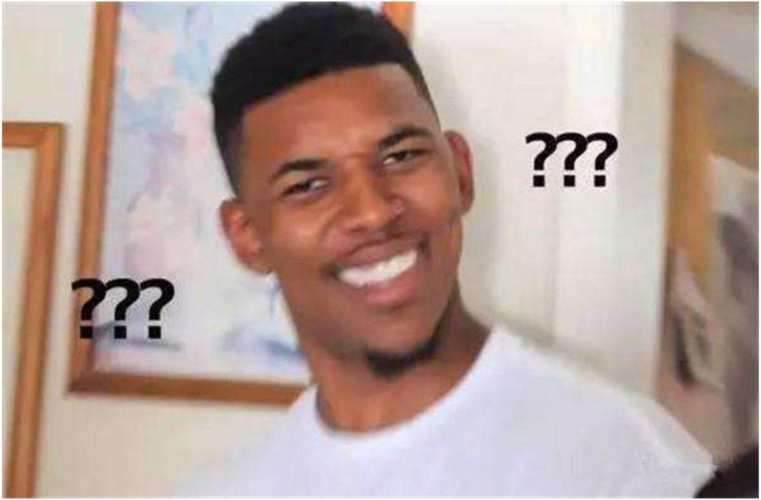
Solution



Laugh Statement

给你一个长度不超过100000的字符串，求出最长的例如*hahahah*或者*ahahahaha*的*ah*相间的子串长度。

Solution



Dna Statement

有一个长度为n的字符串S，只包含0或者1，可惜你不知道S，你每次可以询问一个串P，返回P是否是S的子串，请在不超过t次询问里求出S。

Subtasks

- 1(11 points) $n \leq 5, t = 31$.
- 2(25 points) $n \leq 100, t = 256$.
- 3(64 points) $n \leq 1000, t = 1024$.

Solution for subtask 2

一个容易想到的策略是每次向一个合法的串后面加上0或1，直到不可加，这就可以求出S的一个后缀，再向前不断添加0或1。

这样需要$2n$次询问。

Solution1

沿着上述思路，我们求出后缀以后，向前添加的过程不需要同时判断0和1，因为其中必然有一个是对的。

Solution1

沿着上述思路，我们求出后缀以后，向前添加的过程不需要同时判断0和1，因为其中必然有一个是对的。

我们在求后缀的时候不妨也用同样的方法，每次随机0或1添加，如果连续出现k次错误就认定后缀在这k次之中，然后在其中二分，这样错误率不超过$2^{-k}n$，k只需取15。

这样需要$n + k + \log k$次询问。

Solution2

当然还有不需要随机的做法——先二分出最长全0串的长度l，然后再向后添加求后缀，每次都优先填1，如果连续错误次数超过l就一定错了，再二分即可。

这样需要$n + 2\log n$次询问。

Molecules Statement

有一个含有n个正整数的集合w，给定两个整数l与u，满足

$$u - l \geq \max w_i - \min w_i$$

要求你求出一个w的子集满足其元素之和在$[l, u]$中，或者返回无解。

Subtasks

- 1(09 points) $1 \le n, w_i, l, u \le 100$且所有的$w_i$相同.
- 2(10 points) $1 \le n \le 100, 1 \le w_i, l, u \le 1000, \max w_i - \min w_i \le 1$.
- 3(12 points) $1 \le n \le 100, 1 \le w_i, l, u \le 1000$.
- 4(15 points) $1 \le n \le 10000, 1 \le w_i, l, u \le 10000$.
- 5(23 points) $1 \le n \le 10000, 1 \le w_i, l, u \le 500000$.
- 6(31 points) $1 \le n \le 200000, 1 \le w_i, l, u \le 2^{31}$.

Solution

本题解法多种多样，大家可以随意脑洞。

Solution

本题解法多种多样，大家可以随意脑洞。

这里给出两种方法：先将w排序，一定存在一组连续区间的解或者存在一组由前缀与后缀构成的解。

时间复杂度为$O(n)$或$O(n\log n)$。

Railroad Statement

有n段铁轨，进入第i段铁轨时速度要小于等于s_i，从其中出来时速度变为t_i，初始速度为1，你可以排列这些铁轨，两段铁轨间你可以用1的代价减少1的速度，问你最少需要花费多少代价才可以使火车通过所有的铁轨。

Subtasks

- 1(11 points) $2 \leq n \leq 8$.
- 2(23 points) $2 \leq n \leq 16$.
- 3(30 points) $2 \leq n \leq 200000$.你只需要判断答案是否为0.
- 4(36 points) $2 \leq n \leq 200000$.

对于所有数据$1 \leq s_i, t_i \leq 10^9$.

Solution for subtask 3

先考虑如何判断答案是否为0。

Solution for subtask 3

先考虑如何判断答案是否为0。
贪心？

Solution for subtask 3

先考虑如何判断答案是否为0。

贪心?

显然可以转化为哈密顿回路问题，但这不可解，考虑向欧拉回路上转化。我们认为初始速度无穷小，目标速度无穷大，铁轨就是s_i到t_i的边，我们可以任意加速，也就是添加i到$i+1$的边，通过度数可以递推出这些边的条数，然后判断整个图是否联通即可。

Solution

沿着上面的思路，这下我们可以用一些代价添加$i+1$到i的边，而我们的目标就是让整个图联通，这就是经典的最小生成树问题。

时间复杂度为$O(n\log n)$。

Shortcut Statement

有n个点排成一排，点i和点$i+1$的距离是l_i，点i上还接了一个特殊点记作点i'，和点i的距离为d_i，你现在可以在这n个点中选择两个点连上一条长度为c的边，要使任意两个点的最短距离的最大值尽量小。

Subtasks

- 1(09 points) $2 \le n \le 10$.
- 2(14 points) $2 \le n \le 100$.
- 3(08 points) $2 \le n \le 250$.
- 4(07 points) $2 \le n \le 500$.
- 5(33 points) $2 \le n \le 3000$.
- 6(22 points) $2 \le n \le 100000$.
- 7(04 points) $2 \le n \le 300000$.
- 8(03 points) $2 \le n \le 1000000$.

对于所有数据$1 \le l_i, c \le 10^9, 0 \le d_i \le 10^9$.

Solution for subtask 7

不妨记点l_i的前缀和为len_i。显然只需要考虑特殊点之前的距离,而i'与j'的距离为$len_j - len_i + d_i + d_j$，若在$a$和$b$之间连边，距离为$|len_a - len_i| + |len_b - len_j| + d_i + d_j + c$。

Solution for subtask 7

不妨记点l_i的前缀和为len_i。显然只需要考虑特殊点之前的距离,而i'与j'的距离为$len_j - len_i + d_i + d_j$，若在$a$和$b$ 之间连边，距离为$|len_a - len_i| + |len_b - len_j| + d_i + d_j + c$。

考虑二分答案k然后判断，则若$len_j - len_i + d_i + d_j > k$，需满足$|len_a - len_i| + |len_b - len_j| + d_i + d_j + c \leq k$。

Solution for subtask 7

不妨记点l_i的前缀和为len_i。显然只需要考虑特殊点之前的距离,而i'与j'的距离为$len_j - len_i + d_i + d_j$，若在$a$和$b$ 之间连边，距离为$|len_a - len_i| + |len_b - len_j| + d_i + d_j + c$。

考虑二分答案k然后判断，则若$len_j - len_i + d_i + d_j > k$，需满足$|len_a - len_i| + |len_b - len_j| + d_i + d_j + c \le k$。

不妨将绝对值都拆掉，转化为对$len_a - len_b$与$len_a + len_b$的四个限制，只需要找到最紧的四个限制就可以用two-pointer判断是否有ab满足条件。

Solution for subtask 7

不妨记点l_i的前缀和为len_i。显然只需要考虑特殊点之前的距离,而i'与j'的距离为$len_j - len_i + d_i + d_j$，若在$a$和$b$ 之间连边，距离为$|len_a - len_i| + |len_b - len_j| + d_i + d_j + c$。

考虑二分答案k然后判断，则若$len_j - len_i + d_i + d_j > k$，需满足$|len_a - len_i| + |len_b - len_j| + d_i + d_j + c \leq k$。

不妨将绝对值都拆掉，转化为对$len_a - len_b$与$len_a + len_b$的四个限制，只需要找到最紧的四个限制就可以用two-pointer判断是否有ab满足条件。

而在求限制的时候可以按顺序枚举j用线段树优化，复杂度为$O(n \log^2(nL))$。

Solution

之前算法的瓶颈在于线段树，这里是可以用two-pointer进行优化的，那么复杂度可以降到$O(n\log(nL))$从而通过本题。

Paint Statement

有一个长度为n的黑白序列，告诉你所有k个极长连续黑段长度和顺序。有一些位置的颜色已知，需要判断剩下未知的位置哪些颜色一定是白或一定是黑。保证至少存在一组解。

Subtasks

- 1(07 points) $n \leq 20, k = 1$且所有位置未知.
- 2(03 points) $n \leq 20$且所有位置未知.
- 3(22 points) $n \leq 100$且所有位置未知.
- 4(27 points) $n \leq 100$且已知位置均为白色.
- 5(21 points) $n \leq 100$.
- 6(10 points) $n \leq 5000, k \leq 100$.
- 7(10 points) $n \leq 200000, k \leq 100$.

Solution

因为n很大而k很小，容易猜到最终复杂度为$O(nk)$，于是考虑二维dp。

Solution

因为n很大而k很小，容易猜到最终复杂度为$O(nk)$，于是考虑二维dp。

设$f0[i][j]$表示前i个位置能否匹配j段，并且第i个位置为白，$f1[i][j]$表示前i个位置能否匹配j段，并且第i个位置为黑。这两个dp可以用$O(nk)$的时间算出。

当然也可以从后向前求出$g0$与$g1$。

Solution

因为n很大而k很小，容易猜到最终复杂度为$O(nk)$，于是考虑二维dp。

设$f0[i][j]$表示前i个位置能否匹配j段，并且第i个位置为白，$f1[i][j]$表示前i个位置能否匹配j段，并且第i个位置为黑。这两个dp可以用$O(nk)$的时间算出。

当然也可以从后向前求出$g0$与$g1$。

检查i能否涂白的时候，只需枚举前面有多少段。

Solution

因为n很大而k很小，容易猜到最终复杂度为$O(nk)$，于是考虑二维dp。

设$f0[i][j]$表示前i个位置能否匹配j段，并且第i个位置为白，$f1[i][j]$表示前i个位置能否匹配j段，并且第i个位置为黑。这两个dp可以用$O(nk)$的时间算出。

当然也可以从后向前求出$g0$与$g1$。

检查i能否涂白的时候，只需枚举前面有多少段。

检查涂黑时则稍稍麻烦一点，枚举i能否作为第j段的开头，若可行利用前缀和的技巧给这一段都打上标记。

Messy Statement

这是一道交互题（其实都是交互题）。

交互库维护了一个数据结构，可以存储n位二进制串。一开始你可以向空的数据结构中插入若干二进制串，接下来这个数据结构会将其中存储的二进制串进行改变。

改变的方法是生成一个0到$n-1$的排列p_i，将原来的二进制串$a_0a_1a_2 \ldots a_{n-1}$变成$a_{p_0}a_{p_1}a_{p_2} \ldots a_{p_{n-1}}$。

接着你可以进行询问，每次询问一个串是否在这个数据结构中。

要求你在不超过w次插入和r次询问中求出排列p_i。

Subtasks

- 1(20 points) $n \leq 8, w = 256, r = 256$且只有两个位置$p_i \neq i$.
- 2(18 points) $n \leq 32, w = 320, r = 1024$.
- 3(11 points) $n \leq 32, w = 1024, r = 320$.
- 4(21 points) $n \leq 128, w = 1792, r = 1792$.
- 5(30 points) $n \leq 128, w = 896, r = 896$.

Solution

$w = r = 896 = 128 \times 7 = n\log_2(n)$，提示我们使用分治算法。

Solution

$w = r = 896 = 128 \times 7 = n\log_2(n)$，提示我们使用分治算法。

分治的方法也很容易，对于当前区间$[l, r]$，我们将其分为两部分$[l, mid]$与$[mid + 1, r]$，然后对于$[l, mid]$的每个位置i都插入一个$[l, i - 1]$和$[i + 1, r]$为0的二进制串。那么在询问的时候就可以通过枚举区分出$[l, mid]$与$[mid + 1, r]$并分治下去。

实际上插入只需要一半次数。

Aliens Statement

有一个$m \times m$的正方形网格，其中有n个关键点，你可以花不超过k个小正方形去覆盖这n个点，要求这些小正方形的主对角线必须落在大正方形的主对角线上，并且面积并最小。

Subtasks

- 1(04 points) $1 \le k = n \le 50, 1 \le m \le 100$.
- 2(12 points) $1 \le n \le 500, 1 \le m \le 1000$且关键点都在主对角线上.
- 3(09 points) $1 \le n \le 500, 1 \le m \le 1000$.
- 4(16 points) $1 \le n \le 4000, 1 \le m \le 1000000$.
- 5(19 points) $1 \le n \le 50000, 1 \le k \le 100, 1 \le m \le 1000000$.
- 6(40 points) $1 \le n \le 100000, 1 \le m \le 1000000$.

对于所有数据$1 \le k \le n$.

Solution for subtask 5

为了方便不妨把左下的关键点都折到右上。考虑两个点(x_a, y_a)和(x_b, y_b)，若$x_a \le x_b$且$y_a \ge y_b$时所有覆盖A的正方形均可以覆盖B，则可以删去无用的点，将剩余的点按照横坐标排序以后应该满足$x_i \le x_{i+1}$且$y_i \le y_{i+1}$。

Solution for subtask 5

为了方便不妨把左下的关键点都折到右上。考虑两个点(x_a, y_a)和(x_b, y_b)，若$x_a \leq x_b$且$y_a \geq y_b$时所有覆盖A的正方形均可以覆盖B，则可以删去无用的点，将剩余的点按照横坐标排序以后应该满足$x_i \leq x_{i+1}$且$y_i \leq y_{i+1}$。

这样每个正方形都会覆盖一段连续的点，不妨设$f[i][j]$表示用i个正方形覆盖前j个点时面积并的最小值，转移则是枚举最后一段，即

$$f[i][j] = \min_{0 \leq k < j} \{f[i-1][k] + (y_j - x_{k+1} + 1)^2 - max(0, y_k - x_{k+1} + 1)^2\}$$

Solution for subtask 5

为了方便不妨把左下的关键点都折到右上。考虑两个点(x_a, y_a)和(x_b, y_b)，若$x_a \leq x_b$且$y_a \geq y_b$时所有覆盖A的正方形均可以覆盖B，则可以删去无用的点，将剩余的点按照横坐标排序以后应该满足$x_i \leq x_{i+1}$且$y_i \leq y_{i+1}$。

这样每个正方形都会覆盖一段连续的点，不妨设$f[i][j]$表示用i个正方形覆盖前j个点时面积并的最小值，转移则是枚举最后一段，即

$$f[i][j] = \min_{0 \leq k < j} \{f[i-1][k] + (y_j - x_{k+1} + 1)^2 - max(0, y_k - x_{k+1} + 1)^2\}$$

这个转移满足决策单调性并可以进行斜率优化，复杂度为$O(nk)$。

Solution

这种选取不超过k个的dp题目有一个有趣的做法，如果$f[k][n]$为关于k的凸函数，那就可以用一个一次函数去切这个函数，通过二分其斜率使得切点为k。

在本题中的做法即为不限制选的小正方形数量，而二分一个常数c表示每选一个小正方形就需要付出的代价，直到最优解为k个小正方形。这样的dp只需要一维，并可以进行斜率优化。则复杂度降为$O(n \log nm)$。

Proof

现(bing)在(bu)就(xu)要(yao)证明$f[k][n]$为关于k的凸函数。

Proof

现(bing)在(bu)就(xu)要(yao)证明$f[k][n]$为关于k的凸函数。

我们在序列$(x0, y0), (x1, y1)...(x_{n-1}, y_{n-1})$后添一个点$(Y, Y)$，此时记分$i$段的答案为$Ans(i, Y)$，决策点为$P(i, Y)$，则

$$d(Ans(i, Y))/dY = 2Y - 2x_{P(i,Y)} + 2$$

不难证明$P(i, Y) \le P(i + 1, Y)$，那么

$$d(Ans(i, Y) - Ans(i + 1, Y))/dY \ge 0$$

也就是说$Ans(i, Y) - Ans(i + 1, Y)$随$Y$递增。

Proof

而当$Y \to \infty$时

$$Ans(i, Y) - Ans(i+1, Y) \to f[i-1][n] - f[i][n]$$

当$Y \to y_{n-1}$时

$$Ans(i, Y) - Ans(i+1, Y) \to f[i][n] - f[i+1][n]$$

那么就有$f[i-1][n] - f[i][n] \geq f[i][n] - f[i+1][n]$，这就意味着$f[k][n]$为关于$k$的凸函数。

完结撒花

想要获得原版IOI2016的数据题解题面请戳http://ioinformatics.org/locations/ioi16/contest/index.shtml

祝大家冬令营取得好成绩！ GL&HF！