题解

cxt

August 2, 2018

1 CONNECT

1.1 30 ∼??:

依照题意模拟, 使用 DFS 或 BFS 扩展可到达的点,

暴力 n^2 根据写的是否不满, 以及各种乱搞技巧是否高明, 可以获得大于等于 30 分的成绩

(当然作为一道签到题, 想必大家都会做)

1.2 100:

正解其实就是暴力 BFS.

考虑所有询问子图的边数的和.

设每块分 k 个时和最大, 则此时和为 $min(k*k, m)*(n/k)$

视 n, m 同阶, 则当 k 取 $\sqrt{n}$ 时, 复杂度卡到最高, 为 $n*\sqrt{n}$.

接下来只要设计一个只与原图边数有关的算法.

考虑我们每次 BFS 进行染色. 当我们访问一个未被染色的点, 此时我们要么减少了一条原图中的边, 要么减少了一个原图中的点.

用链表维护未被染色的点即可.

用 hash 判断边是否存在 (没卡 map).

最终复杂度为 $n*\sqrt{n}$.

2 EXPRESSION

2.1 $n \leq 10, p \leq 10$:

数据情况极少, 用于写挂选手…

2.2 保证满足条件:

表面似乎只要输出每个数就好了…
实际上还要判断"-1". "-1" 判法见下文

2.3 只有加号或乘号:

可以用和 (积) 直接计算. 数据中无特殊情况

2.4 没有括号:

同 2.1, 用于写挂选手…

2.5 $n \leq 5000$:

暴力对每一个数枚举它选多少, 然后跑一遍表达式求值判断
注意判断"-1: 和"No Solution", 判法见正解

2.6 100:

考虑对于每一个表达式, 我们都可以建成一颗表达式二叉树.
其每个叶节点都对应原表达式的一个数字. 其每个非叶节点上有一个符号.
性质 1: 中序遍历中将叶节点按顺序取出刚好就是原表达式里的数字.
性质 2: 自底向上计算每个非叶节点的值, 根节点的值即为表达式的值.
于是我们首先用类似于表达式求值的算法构建出这棵表达式树.
然后定义 DFS(x,val) 为当前在 x 点, x 点的值需要为 val 才能使等式成立.
显然每次递归访问孩子时, 可以通过四则运算的逆运算 O(1) 得到两个孩子的 val.
注意预处理逆元 (虽然不预处理应该也跑得过). 总复杂度 O(n)
当处理乘, 除时, 可能会发生 0*x=k 的情况, 根据 k 是否为 0 分别会出现"-1","No Solution", 注意判断.

3 TRAVEL

让我们先不考虑 $P\&Q$

3.1 $k \leq 5$:

似乎正解和暴力都不需要读入 k. k 只是用来保证复杂度的. k 小的时候可以跑得快一些.

3.2 $opt = 1$:

我们只考虑 $P = (1, 0), Q = (0, 1)$ 的情况. 容易发现按 x 坐标排序之后只要求一遍 LIS 即可. 复杂度 $n * \log_2(n)$

3.3 $n \leq 1000$:

那如果有修改呢?
只要每次都重新 sort 一遍再求 lis 即可. 复杂度 $n^2 * \log_2(n)$

3.4 $n \leq 10^5$:

假设只有 3 操作的话, 即每次加入的点 x 坐标都比较大.
我们显然可以加强我们的 LIS 维护方法.
因为用线段树维护 LIS 是可以做到末尾删除的, 所以既然每次只有最多 20 个点要删, 我们就暴力删掉再按顺序加入即可.
那么如果加上 2 操作呢?
其实只要再维护一棵线段树即可. 一棵按 x 下标, 一棵按 y 下标
每次在对应的线段树上修改, 同时把值更新到另一棵线段树上去.
复杂度 $n * 20 * \log_2(n)$

3.5 消除 $P\&Q$:

我们按照每个点在 P, Q 上投影的位置来离散化, 即可给每个点一个新的独一无二的坐标. 注意投影位置相同时, 取坐标小的放在前面.

3.6 100:

结合算法 3.4 以及 3.5 即可. 最终复杂度 $n * 20 * \log_2(n)$