DP题解

本次考试题目难度较为平和，考了几种常见的DP形式。

T1 product

考虑贡献单独计算。

考虑最后出栈的是 i ，则 1 至 i-1 在 i 入栈前就已经弹出，与 i+1 至 n 的顺序没有关系，并且 i+1 至n 的惩罚值只跟他们的顺序与$\Sigma t_j(1 \le j < i)$有关。即可以将 [1,n] 的计算转化为两个子问题 [1,i-1] 和 [i+1,n]。

f[l,r]表示[l,r]的最小惩罚值。

有转移：

$f[l,r] = min(f[l,mid-1] + f[mid+1,r] + (st_{mid-1} - st_{l-1}) * (sd_r - sd_{mid}) + (st_r - st_{l-1}) * d_{mid})$ (l<=mid<=r,其中 s_t 为时间前缀和， s_d为惩罚前缀和)

时间复杂度$O(N^3)$。区间DP常数较小。

空间复杂度$O(N^2)$。

T2

一个显然的结论：假设一个守夜集合为S，则最优代价为$2*(max(x)+max(y))$，即走了一个大矩形。

这启示我们，能够产生贡献的点必然在构成的类似单调栈的结构上。

假设$k=1$,则答案即为$y[1]+x[n]$、

当$k>1$时，此时守夜人会有两种选择：

1、选择一个“垃圾”。“垃圾”定义为单调栈以外的元素。

2、选择若干个栈上元素并吃掉大矩形内的垃圾。

这里又有一个显然的结论：每一个人在单调栈上的选择是连续的。

这样，我们就有一个暴力思路：

设$f[i][k]$为前i个单调栈上的元素，一共选择k次的答案，然后再求一个垃圾的权值排序的和，一一对应合并即可。

复杂度$O(nk)$。

当没有垃圾时，由于答案是凸的（后面会证明），可以wqs二分。复杂度$O(nlogk)$,

考虑另一种思路，我们考虑一个"断点"可以为答案带来如何影响。

假设之前答案为S,则增加量为一个固定的值：$y[i-1]+x[i]$.

我们直接拿堆维护即可。每次取出最大的一个作为新的增量。

这也证明了，答案的增量单调递减，呈凸性可以二分。

这样也可以得到任意个k的答案。

复杂度$O(nlogk)$.

T3

原题是SCOI2015小凸玩密室。

10：暴搜顺序检验。

30：设$f[i][j]$表示访问完第i棵子树，最后留在j号点的最小值。转移显然。由于转移是O(n)的，所以总复杂度O(n^3)。

50：发现一个点只会在LCA处和其他点合并一次，所以上个算法的复杂度其实是均摊O(n^2)的。

特殊：发现dis只跟最后停下的位置有关，而对于一棵子树，最后停下的本质不同的dep只有2种，所以DP是O(n)的。

部分分做法：

考虑n^2的DP：

$f[x][i]$表示访问完当前子树，在i节点结尾的最小代价。

假设先去左边再去右边，转移$f[x][y]=min(f[lc][k]+dis(k,rc)*A[rc]+f[rc][y])$，右儿子同理。

这是当前状态对后续状态仍然有影响的更新方式；但只有$dis(k,u)*A[rc]$这一段是跟k有关的，所以我们改写：

$f[x][y]=dis(y,rc)*A[rc]+f[rc][y]+min(f[lc][k]+dis[k][x]*A[rc])$

我们额外开一个状态$g[x][y]$表示后面那一段即可转移。

注意树是二叉树所以可以O(nlogn)满足。

注：测试数据是只能以1为根，网上代码是可以随意。

T4 report

THUWC 2020 D2T1

~~这题数据也太难造了。~~

$n\leq 15$启发我们状态压缩，一个朴素的想法是只存这个集合决策后得到的最大值和最小值，但是只存最大最小值的话，那么之后的一些以乘法为主的操作可能使得答案向负数的部分偏离。也就是说，最优决策可能需要用一个接近0的数来规避掉一些会使得答案变得不优的转移。

那么转移的时候存最大最小值，和最接近0的正数和负数，但是会发现转移的时候后两个信息没办法正确维护，比方说一个集合能得到的取值可以是$\{-15,-8,-1\}$，然后有一个$(0,0,8)$的操作，这时因为只存了$\{-15,-1\}$这两个数，所以得没办法得到0这个绝对值最小的结果。

注意到转移实际上是$kx+c$的形式，其中$k=0$的话是平凡的，不需要考虑。其他时候有$|k|\geq 1$，所以$|kx|\geq|x|$，也就是说这个乘法部分不会使得答案更加接近0。那么只需要考虑$+c$这部分，这部分如果会出现无法描述且产生更接近0的数只有可能在越过0的转移上出现。为了表示这些转移，可以存下绝对值小于$|nc|$ 的所有位置，这样就可以表示所有可能需要的答案了。

总复杂度是$O(2^nnc^2)$的，实际上跑得非常快。