树形dp&数位dp

题单

chy2011

树形 dp

前置芝士：存树，简单 dp。

一般的思路是，记 $f_{u,\dots}$ 为 u 点你所需要记录的状态。

转移一般就是枚举所有子节点，按一般 dp 的方式进行合并。

但是也有别的情况，比如换根 dp，就是先处理从下往上合并的结果，再处理从上往下更新的结果。

CF1499F

给定一棵 n 个节点的树和一个正整数 k。求有多少种边的集合，使得在将此集合中的所有边断开后，形成的所有连通块的直径都不大于 k。

两个边的集合不同当且仅当存在一条边在一个集合中而不在另一个集合中。

答案对 998244353 取模。

$2 \leq n \leq 5000, 0 \leq k < n$。

CF1499F

显然的思路是：记 $f_{u,i}$ 表示 u 所在的连通块的直径长度为 i 的方案数。

可惜这样不好转移，考虑换一种思路。

记 $f_{u,i}$ 表示 u 所在的连通块以 u 为根的**深度**为 i 的方案数（根的深度为 0）。

转移的时候扫一遍所有子节点枚举 i, j，有两种情况：

CF1499F

1. 切掉 (u,v)，$f_{u,i} \leftarrow f_{u,i} + f_{u,i} \times f_{v,j}$。

2. 不切 (u,v)，此时需要满足 $i+j+1 \leq k$，否则这个连通块的直径会超过 k，$f_{u,\max(i,j+1)} \leftarrow f_{u,\max(i,j+1)} + f_{u,i} \times f_{v,j}$。

代码实现的时候，为了方便，写一个临时转移数组 p，来存储转移后的结果。

可是这样是 $O(nk^2)$ 的，如何优化?

CF1499F

考虑记 h_u 表示当前对 u 来说 f_{u,h_u} 是最大的有方案的 i。转移时只要枚举到 h_u 即可。

根据树形背包的复杂度证明，复杂度为 $O(nk)$。

AT_dp_v

给一棵有 n 个节点的无根树，每个节点只能染成黑色或者白色。对于每一个节点，求出强制把这个节点染成黑色的情况下，所有的黑色节点组成一个联通块的染色方案数，答案对 m 取模。

$1 \leq n \leq 10^5, 2 \leq m \leq 10^9$。

AT_dp_v

显然换根。

考虑以 1 为根，将 u 的答案分为两个部分：u 的父亲方向的方案数（记作 f_u）与 u 的所有儿子方向的方案数（记作 h_u）。答案即为他们的乘积。

对于 f_u，每一个子节点 v 都有两种选择：染黑、染白。对应的方案数分别为 $f_v, 1$。所以 $f_u =$

$$\prod_{v\text{是}u\text{的儿子}} (f_v + 1)$$

这可以用第一次 dfs 来解决。

AT_dp_v

对于 h_u，它的转移关键在于父亲 fa。若父亲染白，方案数为 1，若父亲染黑，放方案数为 h_{fa} 与 u 的兄弟的 f 值 $+1$ 的成绩（每个兄弟都可以染黑或染白），故 $h_u =$

$$(h_{fa} \times \prod_{v\text{是}u\text{的兄弟}} (f_v + 1)) + 1$$

这需要在第二次 dfs 中解决。

AT_dp_v

还有一个问题，h_u 的计算基于 u 的兄弟个数，需要优化到 $O(1)$，故需要在第一次 dfs 中预处理 p_u, s_u 表示 u 之前/之后的兄弟 $f_v + 1$ 乘积。

时间复杂度 $O(n)$。

数位 dp

顾名思义，就是对一个数拆位进行 dp。

一般有两种写法：记搜、递推。

这里我们讲讲记搜。

后文中的 $[X]$ 表示当 X 成立时为 1，否则为 0。

数位 dp

记 $f_{l,\dots}$，表示当前**正在**填第 l 位，后面的状态是什么的方案数（默认不顶上界）。

在记搜的时候，一般写成 `void dfs(int l, ... , bool u)`，最后的 `bool u` 表示是否顶到上界，但在状态中，因为记搜的状态一般可以复用，所以**一般不能**把每次变化的代表上界的 u 写进状态里。

对区间 $[a,b]$ 来说，它的答案可以是 $\mathrm{calc}(b)-\mathrm{calc}(a-1)$，也可以是 $\mathrm{calc}(b)-\mathrm{calc}(a)+[a\text{本身是否是一种情况}]$，其中 $\mathrm{calc}(x)$ 表示 $[1,x]$ 的答案。

数位 dp

下面讲一下数位 dp 的模板写法。

一般将 dp 数组初始化为 -1，因为有的状态答案是 0。

洛谷 P4124

求 $[a,b]$ 范围内的整数中出现至少 3 个相邻的相同数字且号码中不能同时出现 8 和 4 的数字个数。

$10^{10} \leq a \leq b < 10^{11}$。

chy2011

洛谷 P4124

因为 $a-1$ 不一定还是 11 位，我们选择 $\mathrm{calc}(b)-\mathrm{calc}(a)+[a$本身是否是一种情况$]$。

本题中不需要考虑前导 0，因为 l,r 位数相同，所有的因前导 0 而满足的情况都会被抵消。

洛谷 P4124

记 $f_{l,p,P,0/1,0/1,0/1}$ 表示当前正在填第 l 位，前一位的前一位是 p（没有则设为 10，P 同理），前一位是 P，是否已经有三连数，是否有 4，是否有 8 的方案数。

转移十分简单。记忆化搜索的函数为

```
void ll dfs(int l, int p, int P, bool s, bool h4, bool h8, bool u)
```

。

记 g 为这一位的数字上限，则 $g = \begin{cases} n[l], & \text{if } u \\ 9, & otherwise. \end{cases}$

其中 $n[l]$ 表示 n 从右向左数的第 l 位。

chy2011

洛谷 P4124

接着，记 ret 为当前状态答案，将 i 从 0 枚举到 g，每次 $ret \leftarrow ret + \mathrm{dfs}(l-1, p2, i, s \mid\mid [i = p1 = p2], h4 \mid\mid [i = 4], h8 \mid\mid [i = 8], u \;\&\&\; [i = g])$。其中 $\mid\mid$ 是逻辑或，$\&\&$ 是逻辑与。

边界是 $l = 0$ 时，返回 $[s \;\&\&\; !(h4 \;\&\&\; h8)]$，$!$ 表示逻辑非。

存储记忆化时，若 $u = 0$，则 $f_{l,p,P,s,h4,h8} \leftarrow ret$。

调用记忆化时，若 $u = 0$ 且 $f_{l,p,P,s,h4,h8} \neq -1$，则直接返回 $f_{l,p,P,s,h4,h8}$。

然后我们就做完了本题。

洛谷 P4127

给出两个数 a, b，求出 $[a, b]$ 中各位数字之和能整除原数的数的个数。

$1 \leq a \leq b \leq 10^{18}$

chy2011

洛谷 P4127

我们选择 $\mathrm{calc}(b)-\mathrm{calc}(a-1)$。

本题因为前导 0 不影响答案（既不影响数位和，也不影响数字大小），故依然不考虑前导 0。

数位和比较难搞，因为我们不能记录当前数字是多少，只能记录余数。而不知道数位和怎么办？

枚举！

洛谷 P4127

直接枚举当前数位和 c 从 1 到 162（10^{18} 以内的数的数位和上限）

记 $f_{l,s,y}$ 表示当前正在填第 l 位，当前数字和是 s，整个数 $\bmod c = p$ 的方案数。

转移是简单的。$ret \leftarrow ret + \mathrm{dfs}(l-1, s+i, (10y+i) \bmod s, u \;\&\&\; [i=g])$。

剩余细节请自行实现。

Gym 102900C

计算

$$\left(\sum_{i=0}^{X}\sum_{j=[i=0]}^{Y}[i\&j=0]\lfloor\log_2(i+j)+1\rfloor\right) \bmod 1,000,000,007$$

& 是按位与。

$0 \le X, Y \le 10^9$。

chy2011

Gym 102900C

首先将所有数转为二进制计算。

原式相当于求所有 $0 \le a \le X, 0 \le b \le Y$，且 a, b 无相同位皆为 1 ，他们的和的首位位数之和。

这个东西不是很好统计，先考虑如何求出方案数，这是非常简单的，记 $f_{l,ua(0/1),ub(0/1)}$ 为当前在填第 l，a 是否顶到上界，b 是否顶到上界的方案数，因为有两个数，此时我们需要将 ua, ub 传进 dfs 中，这一部分非常简单，请自行实现。

Gym 102900C

考虑如何统计答案。

我们在记忆化的时候已经求出对于所有 l, ua, ub 的方案数，我们可以很好的利用这一点。枚举所有二进制位作为和的首位（也就是答案），计算对应的方案数，相乘求和即可。

假如说将 a 的第 k 位选了 1，则 b 的第 k 位为 0，此时 $ua = [a < 2^{k+1}], ub = [b < 2^k]$，方案数为 $f_{l,ua,ub}$，答案增加 $k \times f_{l,ua,ub}$，b 的第 k 位选 1 是对称的。

hdu 4352

定义 $f(x)$ 为将 x 按位分为一个自然数序列的 LIS 长度（x 不能有前导 0），给定 a, b, k，求 $\sum_{i=a}^{b}[f(i) = k]$。

$0 < a \leq b < 2^{63} - 1, 1 \leq k \leq 10$。

hdu 4352

显然数位 dp。

考虑你需要记录什么状态。

我们要求的是 LIS，参考 LIS 的 $O(n \log n)$ 做法，记录在 LIS 数组内的元素（状态压缩）。

比如你记录的是 $1, 3$ 出现了，那么我们就相当于表示长度为 1 的 LIS 结尾最小为 1，长度为 2 的 LIS 结尾最小为 3。

hdu 4352

假设你现在又填了一个数 p，那么按照朴素的 LIS 的 $O(n \log n)$ 做法，我们直接从 p 向后枚举，找到第一个 $\geq p$ 的出现位置，把他替换成 p，若无法找到，则新增一个 p 出现即可。

接下来就是普通的数位 dp 了，这里讲一下如何处理前导 0：在 dfs 的时候多记一个 `bool f0` 表示现在是否还在填前导 0，只有填完之后不再填前导 0 了才对 LIS 状态进行转移。

hdu 4352

大致代码如下：

```
int zy(int S, int i)
{
    int j; bool fl = 0;
    for (j = i; j <= 9; j++) if (S & (1 << j)) {fl = 1; break;}
    if (fl) S = (S ^ (1 << j)) | (1 << i);
    else S = S | (1 << i);
    return S;
}
ll dfs(int l, int S, bool u, bool f0)
{
    if (!l) return __builtin_popcount(S) == k;
    if (!u && f[l][S][k] != -1) return f[l][S][k];
    for (int i = ; i <= ; i++)
    {
        int nS = zy(S, i);
        ret += dfs(l - 1, (f0 && !i) ? 0 : nS, u && (i == g), f0 && !i);
    }
    if (!u) f[l][S][k] = ret;
    return ret;
}
```

谢谢大家

chy2011