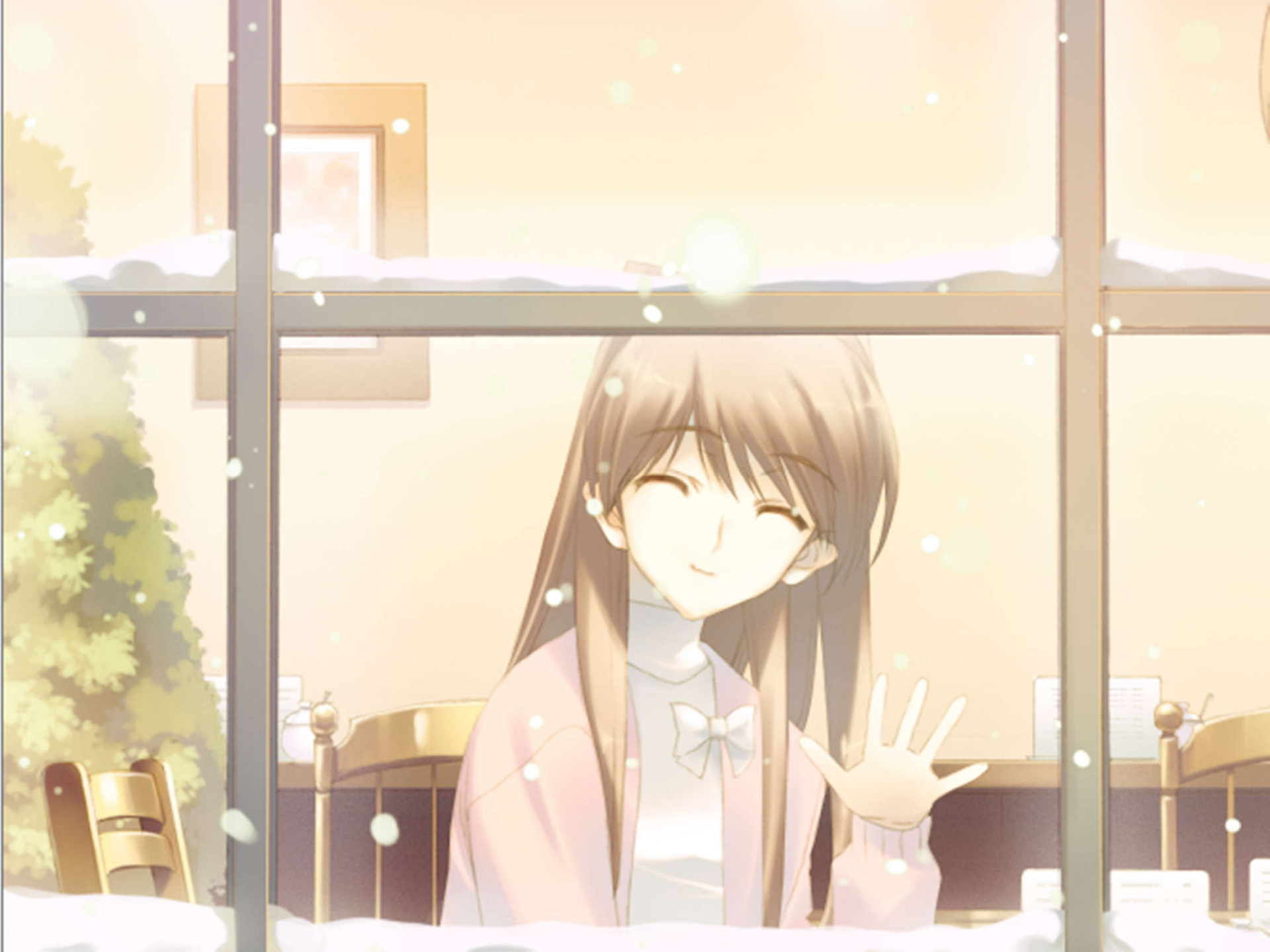
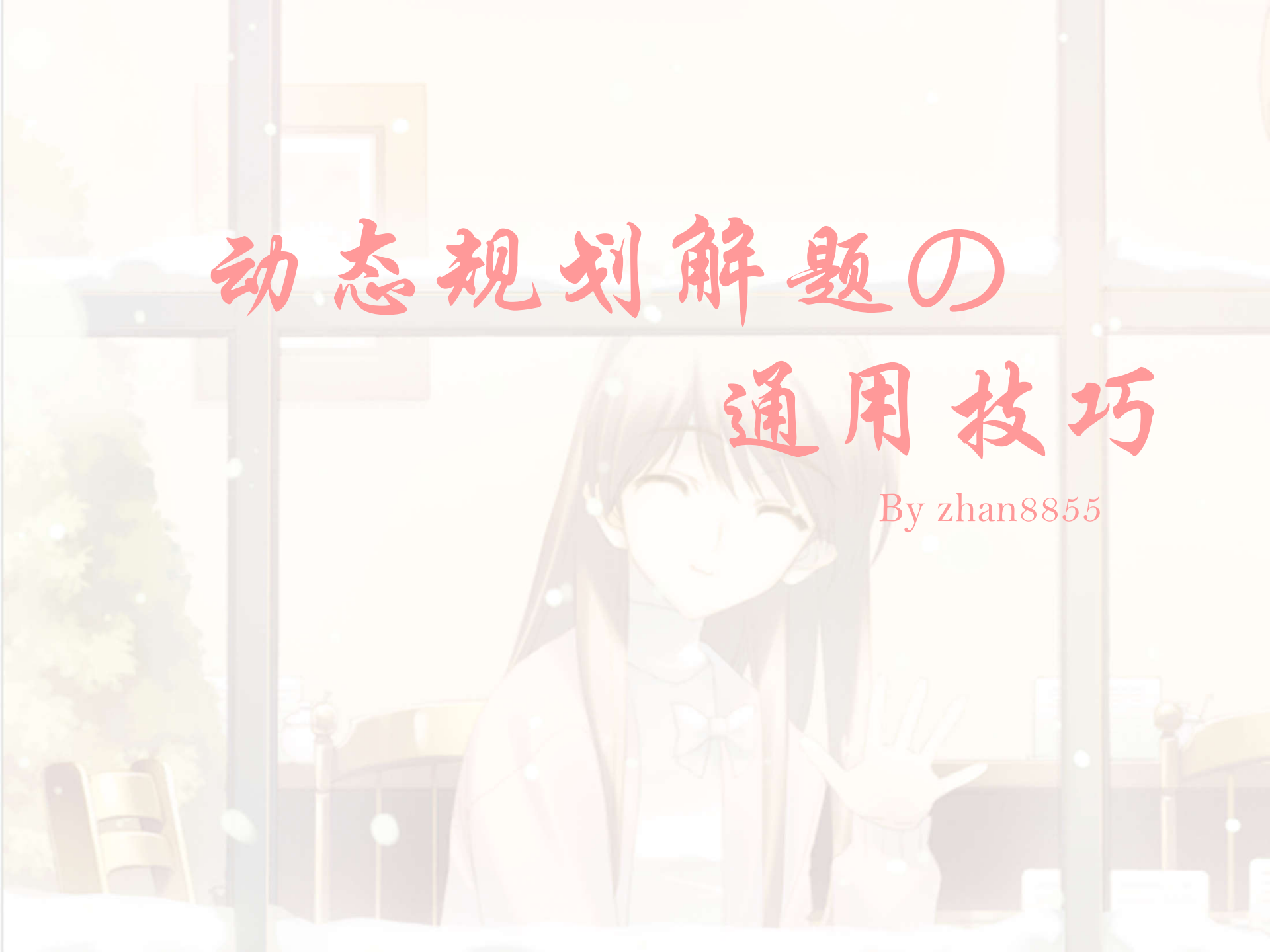


动态规划解题の通用技巧

By zhan8855



目录

1 状态的设计

- ✓ 要尽量利用题目允许的时间和空间，尽可能清晰地实现。
- ✓ 要用尽量少的信息清晰地描述状态，求精求简。
- ✓ 要便于方程的转移。

例题

[ZJOI2008] 生日聚会

题意：

求由n个0、m个1组成，且满足任意子串0的数量和1的数量差绝对值不超过k的01串数量。n, m≤150，k≤20。

例题

[ZJOI2008] 生日聚会

题意：

求由n个0、m个1组成，且满足任意子串0的数量和1的数量差绝对值不超过k的01串数量。n, m≤150，k≤20。

标准算法：

f[i][j][k][l]表示i个0、j个1组成的01串前缀、其所有后缀1的数量减去0的数量最大值为k、最小值为-l的方案数。

例题

[Codeforces814E] An unavoidable detour for home

题意：

求有多少个N个点的带标号无向图，满足以下三条限制：

①所有点到1号点的最短路唯一确定；

②第i个点到1号点的最短路长度不超过第i+1个点到1号点的最短路；

③所有点的度数均给出，且在[2,3]内。

N≤50。

例题

[Codeforces814E] An unavoidable detour for home

标准算法一：

为了满足第二个条件，可以发现，1——N分段分别打乱后要求可以构成图的从1号点开始的BFS序。在此基础上为了满足第一个条件，BFS生成树不存在返祖边。接下来就只有度数的问题了。

用F[i][j][k]表示生成树的前几层共有i个点，其中最后一层有j个点度数还差1，k个点度数还差2。可以用组合数计算出最后一层向下一层连树边的方案数，接下来就只有横叉边的问题了。

直接在动态规划转移同时处理横叉边会影响复杂度。

例题

[Codeforces814E] An unavoidable detour for home

标准算法一：

考虑预处理G[i][j][k][l]表示i个度差1的点和j个度差2的点连横叉边结果变成了k个度还差1的点和l个度还差2的点的方案数。实现时使用顺推比较方便，转移G时先放所有度还差2的点，再放所有度还差1的点，避免重复。转移F时枚举k和l。

总时间复杂度O(N^5)，空间复杂度O(N^4)。

例题

[Codeforces814E] An unavoidable detour for home

标准算法二：

用F[i][j]表示生成树的前i个点分为若干层，其中最后一层的点数有j个的方案数。用辅助数组G[i][j][k]表示生成树本层有i个点，上一层有j个点度为2、k个点度为3的方案数。明显，F[i][j]=∑F[i-j][k]×G[j][c2][c3]。其中c2、c3可以直接由k计算得到。

例题

[Codeforces814E] An unavoidable detour for home

标准算法二：

用F[i][j]表示生成树的前i个点分为若干层，其中最后一层的点数有j个的方案数。用辅助数组G[i][j][k]表示生成树本层有i个点，上一层有j个点度为2、k个点度为3的方案数。明显，F[i][j]=∑F[i-j][k]×G[j][c2][c3]。其中c2、c3可以直接由k计算得到。

考虑转移G。首先，G[0][0][k]意味着上一层的若干个度为3的点形成了若干个环，可以枚举编号最大的点所在的环长，结合项链数(N-1)!/2来计算（除去循环的N种和对称的2种）。接着再依次考虑G[0][j][k]和G[i][j][k]。

总时间复杂度O(N^3)，空间复杂度O(N^3)。

例题

[SCOI2005] 最大子矩阵

题意：

在2×N的矩阵中求K个不重叠的子矩阵，使子矩阵之和最大。N≤100, K≤10。

例题

[SCOI2005] 最大子矩阵

题意：

在2×N的矩阵中求K个不重叠的子矩阵，使子矩阵之和最大。N≤100, K≤10。

标准算法：

动态规划数组前两维表示已经处理到的位置和放入的矩形数。第三维分5种情况：0表示当前行两个数都不选，没有加入矩阵；1表示当前行选左侧的数，加入一个矩阵；2表示当前行选右侧的数，加入一个矩阵；3表示当前行两个数都选，加入一个矩阵；4表示当前行两个数都选，加入两个矩阵。

2 转移的设计

- ✓ 要考虑周到，保证动态规划的正确性。
- ✓ 要充分利用状态的定义，用尽可能简单的状态和方程进行转移。
- ✓ 与状态的设计相辅相成。

例题

[Codeforces392B] Tower of Hanoi

题意：

在传统的汉诺塔游戏上，给出d[i][j]为将一个盘子从i移到j的代价，求将N个盘子从1号柱移到3号柱的最小代价。

N≤40。

例题

[Codeforces392B] Tower of Hanoi

题意：

在传统的汉诺塔游戏上，给出d[i][j]为将一个盘子从i移到j的代价，求将N个盘子从1号柱移到3号柱的最小代价。N≤40。

注意：

要分清题目对不同转移方式的限制。在转移第一层的时候可以用到最短路，因为最小的盘子可以随时放在任何地方；而后来就不行了，因为必须保证大的盘子不能压在小的盘子上面。

例题

[Codeforces722E] Research Rover

题意：

给出一个N*M的方格阵，从(1,1)出发，到(N,M)结束，从(x,y)只能走到(x+1,y)或(x,y+1)。方格阵上还有K个特殊点，初始时给出的分数t每经过一个特殊点就会变成ceil(t/2)。求到(N,M)时得分的期望。保证(1,1)和(N,M)不是特殊点。

N,M≤100000， K≤2000， t≤1000000。

例题

[Codeforces722E] Research Rover

标准算法：

N,M的范围较大，不考虑将其记入动态规划状态。把(1,1)和(N,M)也看成特殊点，把所有的特殊点按照横坐标排序，F[i][j]表示从(1,1)出发到达第i个特殊点、已经经过j个特殊点的方案数。明显，j的取值范围只有log级别。

例题

[Codeforces722E] Research Rover

标准算法：

N,M的范围较大，不考虑将其记入动态规划状态。把(1,1)和(N,M)也看成特殊点，把所有的特殊点按照横坐标排序，F[i][j]表示从(1,1)出发到达第i个特殊点、已经经过j个特殊点的方案数。明显，j的取值范围只有log级别。

考虑转移，F[i][j]=F[k][j-1]*(D[k][i]-∑F[l][j-1]*E[l][i])，其中D[i][j]表示第i个特殊点到第j个特殊点的路径数，可以直接用组合数计算；E[i][j]表示第i个特殊点到第j个特殊点不经过任何其他特殊点的路径数，可以O(K^3)预处理，成为复杂度的瓶颈。

例题

[Codeforces722E] Research Rover

标准算法：

N,M的范围较大，不考虑将其记入动态规划状态。把(1,1)和(N,M)也看成特殊点，把所有的特殊点按照横坐标排序，F[i][j]表示从(1,1)出发到达第i个特殊点、已经经过j个特殊点的方案数。明显，j的取值范围只有log级别。

考虑转移，F[i][j]=F[k][j-1]*(D[k][i]-∑F[l][j-1]*E[l][i])，其中D[i][j]表示第i个特殊点到第j个特殊点的路径数，可以直接用组合数计算；E[i][j]表示第i个特殊点到第j个特殊点不经过任何其他特殊点的路径数，可以O(K^3)预处理，成为复杂度的瓶颈。

更换转移的思路，F[i][j]=D[1][i]-∑F[k][j](k<i)-∑F[i][l](l<j)。时间复杂度降为O(K^2 log t)。

2 转移的设计

动态规划计数去重的一般思路：

- ✓ 容斥原理
- ✓ 分类讨论
- ✓ 最小表示
- ✓ 重设状态

例题

[BZOJ3769] BST Again

题意：

求深度为H节点数为N的不同形态二叉树数目。N, H≤600。

例题

[BZOJ3769] BST Again

题意：

求深度为H节点数为N的不同形态二叉树数目。N, H≤600。

标准算法：

f[i][j]表示j个点搭i层树的方案总数，s[j]表示j个点搭低于i层树的方案总数。

可以写出如下方程：

$f[i][j]=\sum(f[i-1][k]*s[j-k-1]+f[i-1][j-k-1]*s[k])$

但是，这个方程有着可怕的重复计算问题，去重工作任重道远。

例题

[BZOJ3769] BST Again

题意：

求深度为H节点数为N的不同形态二叉树数目。N, H≤600。

标准算法：

f[i][j]表示j个点搭i层树的方案总数，s[j]表示j个点搭低于i层树的方案总数。

不妨分类讨论是否两边都搭到i-1层：

$f[i][j]=\sum(2*(s[k]-f[i-1][k])*f[i-1][j-k-1])+\sum(f[i-1][k]*f[i-1][j-k-1])$

这样就可以通过本题了。

3 方程的特点和优化

3.1 决策点的单调性
3.2 对象的独立性
3.3 转移的单一性
3.4 方程的冗余性
3.5 预处理技巧
3.6 常数优化策略

3.1 决策点的单调性

- ✓ 如果观察状态转移方程得到决策点关于状态的单调性关系，就可以对方程进行一个比较显著的优化，通常可以降低一维时间复杂度。

例题

[USACO 2007 November Gold] Telephone Wire

题意：

有N根电线杆，要给相邻的两根连线。可以选择拔高其中一部分电线杆，拔高一根电线杆操作的代价为△H^2。拔高后，要给相邻的电线杆连线，连线相邻两根电线杆操作的代价为C*△H，其中C是一个常量且已经给出。求最小总代价。

例题

[USACO 2007 November Gold] Telephone Wire

标准算法一：

f[i][j]表示第i根电线杆高度为j需要的最小总代价，于是可以得到：

f[i][j]=min{f[i-1][k]+C*abs(j-k)}+(h[i]-j)^2

这个方程可以加一些简单的剪枝。首先，j和k不必要从1开始枚举，直接从h[i]和h[i-1]开始枚举即可；而且，维护s[i]表示前i根电线杆连线的最小总代价，转移时如果s[i-1]+C*abs(j-k)+(h[i]-j)^2>f[i][j]可以直接终止循环。时间复杂度O(NH^2)。

例题

[USACO 2007 November Gold] Telephone Wire

标准算法一：

f[i][j]表示第i根电线杆高度为j需要的最小总代价，于是可以得到：

f[i][j]=min{f[i-1][k]+C*abs(j-k)}+(h[i]-j)^2

这个方程可以加一些简单的剪枝。首先，j和k不必要从1开始枚举，直接从h[i]和h[i-1]开始枚举即可；而且，维护s[i]表示前i根电线杆连线的最小总代价，转移时如果s[i-1]+C*abs(j-k)+(h[i]-j)^2>f[i][j]可以直接终止循环。

时间复杂度O(NH^2)。

标准算法二：

对于递增的j，最优的k值也必然是递增的。因此，可以利用这个单调性来求解。

时间复杂度O(NH)。

例题

[USACO 2007 November Gold] Telephone Wire

标准算法三：

假设已经枚举出所有不进行拔高的电线杆，不拔高的电线杆把序列划分为若干个区间，要求对所有其他电线杆进行拔高操作，那么最优方案必定是对每个区间分别设定一个高度，把区间内所有的电线杆都拔到这个高度。

观察可以发现每个区间的代价都是一个二次函数，可以O(1)求出极值。

同时可以发现，如果区间中最高的电线杆不低于区间端点之一，将整个区间拔高并不优。因此，可以用单调栈维护动态规划求解。

时间复杂度O(N)。

3.1 决策点的单调性

- ✓ 区间型动态规划的合并点若满足单调性，就可以由此入手优化动态规划。这种方法称为四边形不等式。
- ✓ 但是，有时候直接观察或许并不能得到结果。可以使用确定左端点、观察决策点随右端点的变化来确定思路。

例题

[2016Multi-UniversityTrainingContest 5A] ATM Mechine

题意：

去ATM机取款，每次输入取款额，如果该额不超过存款额，ATM机就会吐出货币；否则，ATM机会警告。如果警告的次数达到K次，它会响警铃。现在要把钱取完，且不能让它响铃，求期望的操作数。已知存款总额在[0..N]中等概率分布。N, K≤2000。

例题

[2016Multi-UniversityTrainingContest 5A] ATM Mechine

题意：

去ATM机取款，每次输入取款额，如果该额不超过存款额，ATM机就会吐出货币；否则，ATM机会警告。如果警告的次数达到K次，它会响警铃。现在要把钱取完，且不能让它响铃，求期望的操作数。已知存款总额在[0..N]中等概率分布。N, K≤2000。

标准算法：

动态规划。f[i][j]表示存款在[0..j]时ATM机还可以警告i次的期望操作数。则有方程：

f[i][j]=(f[i-1][k-1]*k+f[i][j-k]*(j-k+1))/(j+1)+1;

其中k是当前输入的取款额。易得这个额值的选择随j的增长而单调递增。

3.1 决策点的单调性

- ✓ 对于较复杂的决策点满足单调性的情况，可以采用二分的方式确定每个决策点对应的决策区间。

例题

[POI2011] Lightning Conductor

题意：

已知一个长度为n的序列G，要求对于每个元素，找到最小的非负整数p，满足对于任意的j，Gj≤Gi+p-sqrt(abs(i-j))。

n≤500000。

例题

[POI2011] Lightning Conductor

题意：

已知一个长度为n的序列G，要求对于每个元素，找到最小的非负整数p，满足对于任意的j，Gj≤Gi+p-sqrt(abs(i-j))。

n≤500000。

标准算法：

移项得Gi+p≥Gj+sqrt(abs(i-j))。题意转化为对于每个位置i，求Gj+sqrt(abs(i-j))的最大值。

为了排除abs对转移的干扰，可以强制i<j做一遍，再强制i>j做一遍，之后取最大值。观察后发现决策点满足单调性。可以用单调栈维护决策点，对于每个新插入的决策点，二分出它对应的决策区间，将其加入单调栈。

3.1 决策点的单调性

- ✓ 斜率优化是另一种形式的决策点单调性的利用，适用于类似直线形式的动态规划方程。
- ✓ 一般思路是把问题转化为直线集合在某一点的最值，用凸壳来维护。

例题

[Codeforces455E] Function

题意：

对于长度为N的数列g[j]，令f[1][j]=g[j]，
f[i][j]=min(f[i-1][j],f[i-1][j-1])+g[j]。
M组询问，求f[x][y]的值。
x≤y≤N≤100000，M≤100000。

例题

[Codeforces455E] Function

题意：

对于长度为N的数列g[j]，令f[1][j]=g[j]，
f[i][j]=min(f[i-1][j],f[i-1][j-1])+g[j]。
M组询问，求f[x][y]的值。
x≤y≤N≤100000，M≤100000。

标准算法：

题目给出的问题等价于一个如下的问题：在y-i+1到y中选x个数，这x个数中必须有g[y]，且必须连续。求选出数字的最小和。

例题

[Codeforces455E] Function

题意：

对于长度为N的数列g[j]，令f[1][j]=g[j]，
f[i][j]=min(f[i-1][j],f[i-1][j-1])+g[j]。
M组询问，求f[x][y]的值。
x≤y≤N≤100000，M≤100000。

标准算法：

题目给出的问题等价于一个如下的问题：在y-i+1到y中选x个数，这x个数中必须有g[y]，且必须连续。求选出数字的最小和。
进一步分析答案=g[l]*(x-(y-l+1))+(s[y]-s[l-1])，其中s是g的前缀和。将其化为g[l]*(x-y-1)+g[l]*l-s[l-1]+s[y]。
这是直线的方程。

例题

[Codeforces455E] Function

题意：

对于长度为N的数列g[j]，令f[1][j]=g[j]，
f[i][j]=min(f[i-1][j],f[i-1][j-1])+g[j]。
M组询问，求f[x][y]的值。
x≤y≤N≤100000，M≤100000。

标准算法：

注意到当x不变时，决策点随y是单调的。当y不变时，决策点随x是单调的。
考虑用一个数据结构维护决策点直线的凸壳，对于询问在数据结构中二分查找，每次把相邻两条直线进行比较。

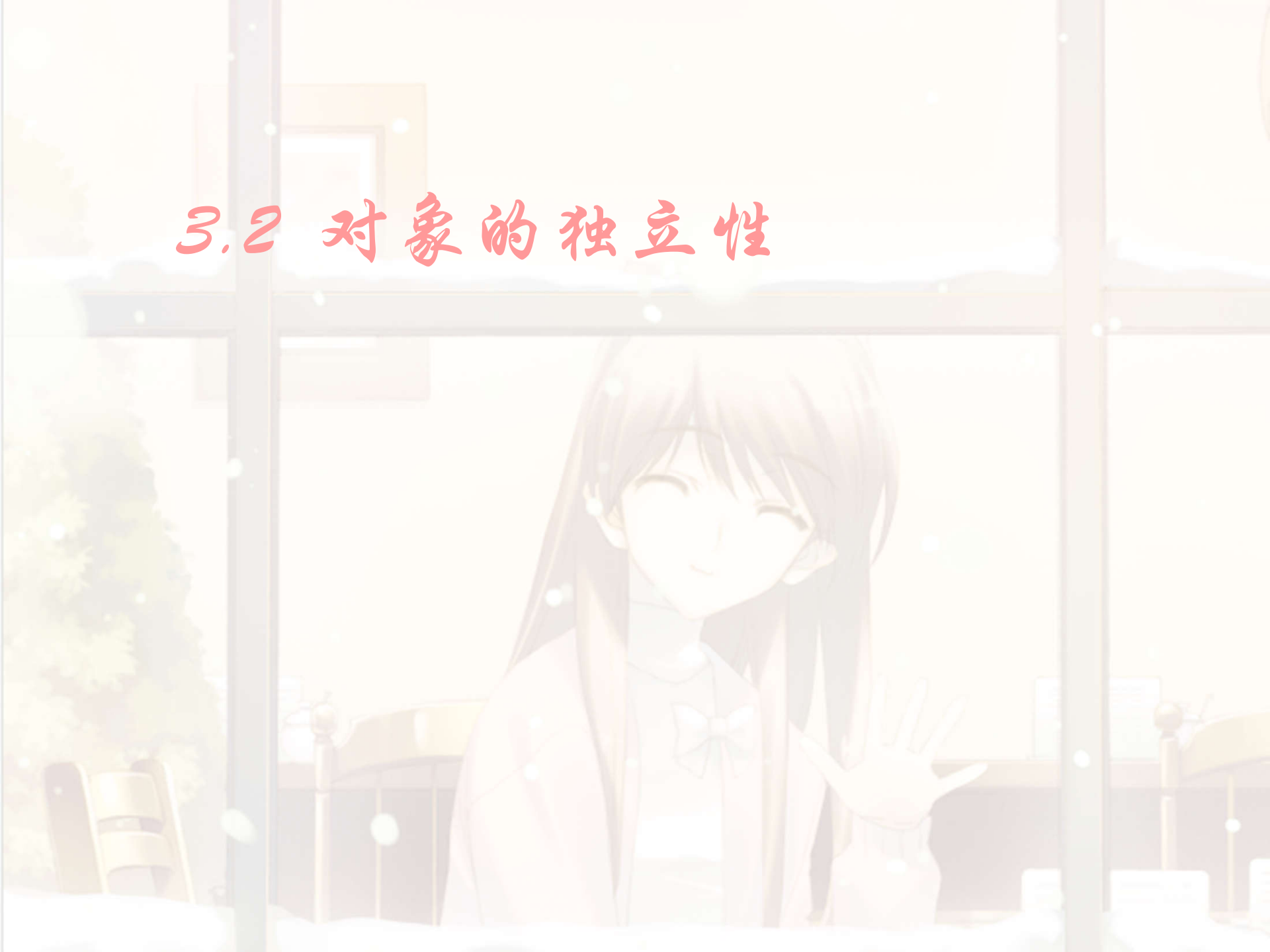
3.1 决策点的单调性

- ✓ 如果决策点不满足单调性或者决策点有多个，但是转移方程中的一些变量满足单调性，方程可以用高级数据结构（如线段树）来维护。

基础练习

[USACO 2008 March Gold] Land Acquisition
[USACO 2009 Open Gold] Tower of Hay
[USACO 2011 Feburary Gold] Generic Cow Protests
[USACO 2011 Open Gold] Mowing the Lawn

3.2 对象的独立性



例题

[Codeforces712D] Memory and Scores

题意：

两个人玩游戏，共进行t轮，每人每轮从[-k,k]中选出一个数字，将其加到自己的总分中。已知两人的初始得分分别为a和b，求第一个人最后获胜的方案数。两种方案被认为是不同的，当且仅当存在其中一轮，其中一人选到的数字不同。a, b, t≤100，k≤1000。

例题

[Codeforces712D] Memory and Scores

题意：

两个人玩游戏，共进行t轮，每人每轮从[-k,k]中选出一个数字，将其加到自己的总分中。已知两人的初始得分分别为a和b，求第一个人最后获胜的方案数。两种方案被认为是不同的，当且仅当存在其中一轮，其中一人选到的数字不同。a, b, t≤100，k≤1000。

标准算法：

可以发现，两个人的操作互不干扰。令f[i]表示一个人玩游戏加了i分的方案数，动态规划即可。

3.2 对象的独立性

- ✓ 对于相互独立的对象，分别计算往往比同时计算效率更优秀。

例题

[BestCoder Round #86 1004] Keep In Touch

题意：

有三个人从一张N个点无重边的有向无环图上的三个点出发，每单位时间，他们分别选择当前点的一条出边走下去。有向无环图点有点权，任意时刻他们所在的三个点两两点权相差不超过K。他们可以在任意三个点同时结束。求合法的路径总数。N≤50。

例题

[BestCoder Round #86 1004] Keep In Touch

题意：

有三个人从一张N个点无重边的有向无环图上的三个点出发，每单位时间，他们分别选择当前点的一条出边走下去。有向无环图点有点权，任意时刻他们所在的三个点两两点权相差不超过K。他们可以在任意三个点同时结束。求合法的路径总数。N≤50。

普通算法：

f[i][j][k]表示第一个人在i、第二个人在j、第三个人在k的方案总数，可以枚举f[x][y][z]转移过来。总复杂度O(N^6)。

例题

[BestCoder Round #86 1004] Keep In Touch

题意：

有三个人从一张N个点无重边的有向无环图上的三个点出发，每单位时间，他们分别选择当前点的一条出边走下去。有向无环图点有点权，任意时刻他们所在的三个点两两点权相差不超过K。他们可以在任意三个点同时结束。求合法的路径总数。N≤50。

标准算法：

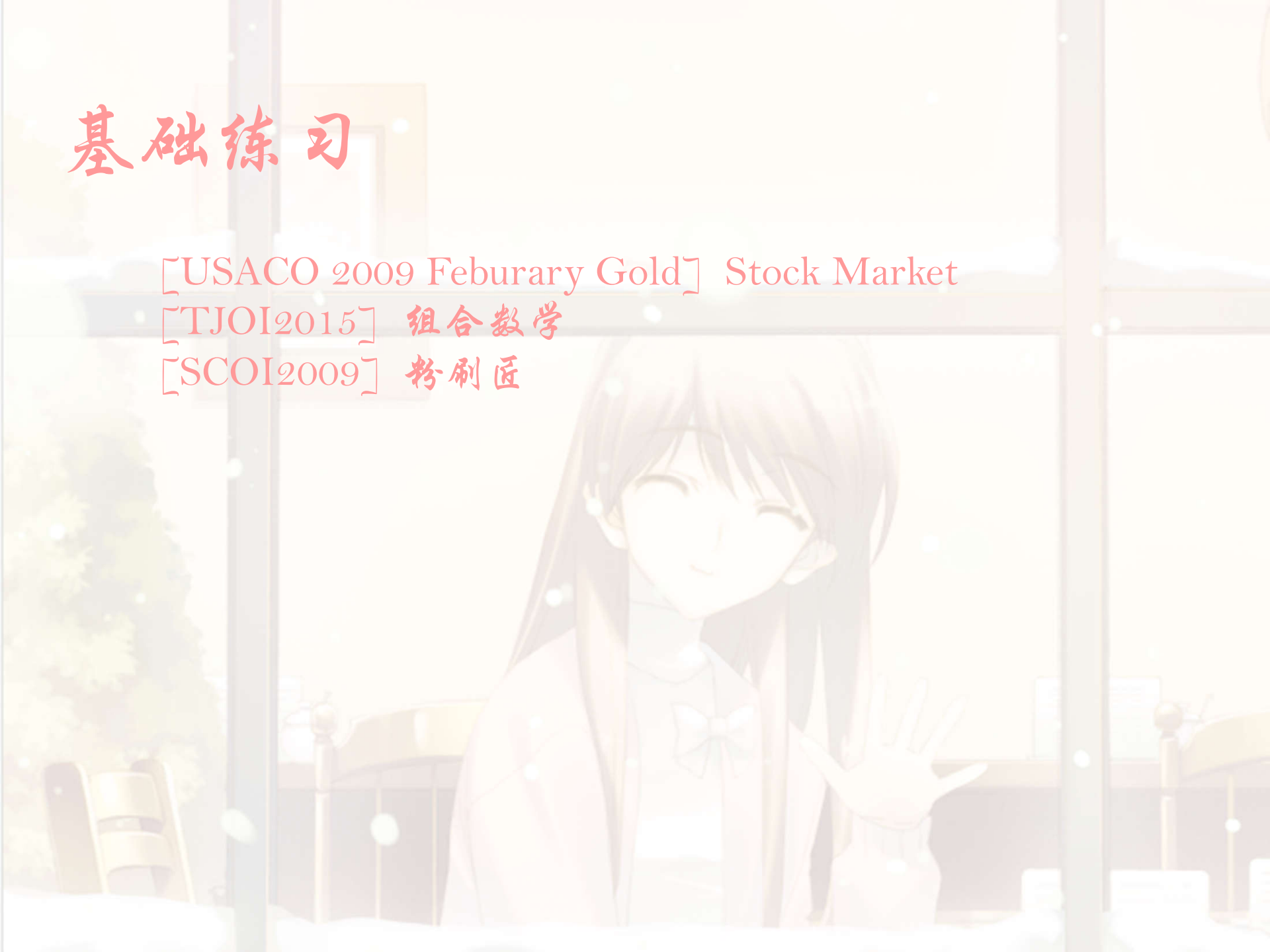
考虑到每个人每一步的选择都是独立的，所以让三个人可以分别走。增加一维状态，f[i][j][k][l]表示第一个人在i，第二个人在j，第三个人在k，当前是第l个人走的方案数，转移复杂度降到O(N)，总复杂度降到O(N^4)。

基础练习

[USACO 2009 Feburary Gold] Stock Market

[TJOI2015] 组合数学

[SCOI2009] 粉刷匠



3.3 转移的单一性

- ✓ 根据某些转移不变的性质，和时空允许的条件，可以使用矩阵乘法优化方程式。

例题

[Codeforces750E] New Year and Old Subsequence

题意：

给出一个长度为N的数字串，M次询问其中一个区间至少要删去多少数字，才能满足“2016”不是它的子序列而“2017”是它的子序列。N, M≤200000。

例题

[Codeforces750E] New Year and Old Subsequence

题意：

给出一个长度为N的数字串，M次询问其中一个区间至少要删去多少数字，才能满足“2016”不是它的子序列而“2017”是它的子序列。N, M≤200000。

标准算法：

考虑用线段树维护动态规划数组。设立五个状态“”、“2”、“20”、“201”、“2017”。如果运用区间型动态规划，则会出现较难处理的后效性。只有线型动态规划才能较好地解决这个问题。

例题

[Codeforces750E] New Year and Old Subsequence

题意：

给出一个长度为N的数字串，M次询问其中一个区间至少要删去多少数字，才能满足“2016”不是它的子序列而“2017”是它的子序列。N, M≤200000。

标准算法：

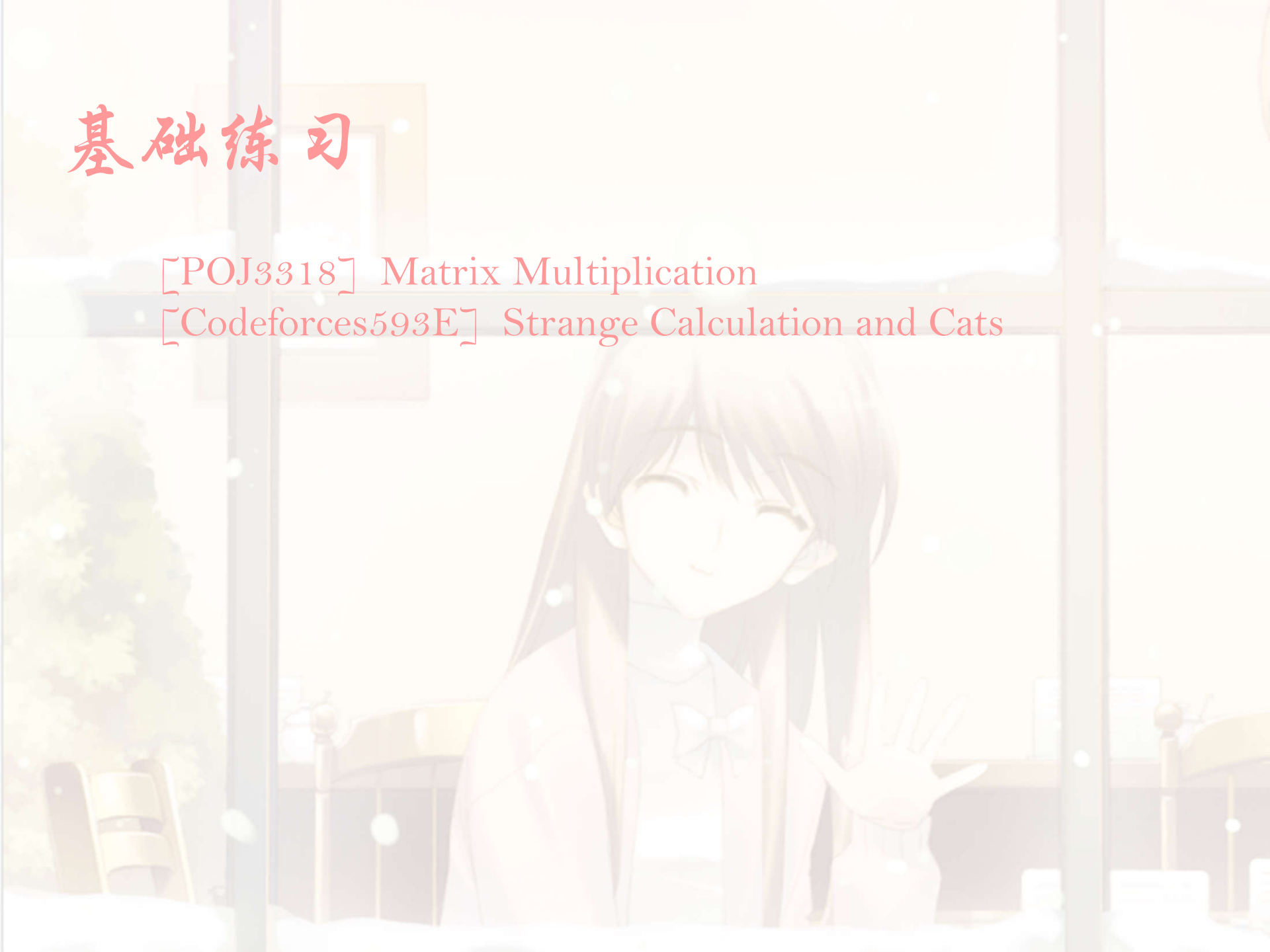
考虑用线段树维护动态规划数组。设立五个状态“”、“2”、“20”、“201”、“2017”。如果运用区间型动态规划，则会出现较难处理的后效性。只有线型动态规划才能较好地解决这个问题。

因此，构建出线型动态规划的初始矩阵和转移矩阵，线段树维护矩阵即可。

基础练习

［POJ3318］Matrix Multiplication

［Codeforces593E］Strange Calculation and Cats



3.4 方程的冗余性

- ✓ 有些方程虽然比较复杂，但是实际有效的状态和转移有限，经过挖掘，也许就可以发现优化的策略。

例题

[USACO 2009 March Gold] Cleaning Up

题意：

把长度为N的数列分成任意多段，每段数列的不和谐度为该段内不同数字数量的平方，总不和谐度为所有段的不和谐度之和，求总不和谐度的最小值。N≤40000。

例题

[USACO 2009 March Gold] Cleaning Up

题意：

把长度为N的数列分成任意多段，每段数列的不和谐度为该段内不同数字数量的平方，总不和谐度为所有段的不和谐度之和，求总不和谐度的最小值。N≤40000。

标准算法：

可以写出这样一个方程：f[i]=min{f[j-1]+color[j,i]^2}

其中color[j,i]表示区间[j,i]内不同数字数量。

例题

[USACO 2009 March Gold] Cleaning Up

题意：

把长度为N的数列分成任意多段，每段数列的不和谐度为该段内不同数字数量的平方，总不和谐度为所有段的不和谐度之和，求总不和谐度的最小值。N≤40000。

标准算法：

可以写出这样一个方程：f[i]=min{f[j-1]+color[j,i]^2}

其中color[j,i]表示区间[j,i]内不同数字数量。

进一步思考可以发现，如果把每个数分一段，总不和谐度即为数列长度；也就是说，答案最终必定不会超过数列长度。所以，方程中color[j,i]最大不能超过根号级别。

例题

[USACO 2009 March Gold] Cleaning Up

题意：

把长度为N的数列分成任意多段，每段数列的不和谐度为该段内不同数字数量的平方，总不和谐度为所有段的不和谐度之和，求总不和谐度的最小值。N≤40000。

标准算法：

可以写出这样一个方程：f[i]=min{f[j-1]+color[j,i]^2}

其中color[j,i]表示区间[j,i]内不同数字数量。

进一步思考可以发现，如果把每个数分一段，总不和谐度即为数列长度；也就是说，答案最终必定不会超过数列长度。所以，方程中color[j,i]最大不能超过根号级别。

用一个长度为根号的队列，第k个元素维护当前位置往前出现不超过k种颜色能够延伸到的位置即可。

例题

[Codeforces729F] Financiers Game

题意：

两个人分别从长度为n的数列的两端开始取数，如果前一个人取了k个数，后一个人必须取k或k+1个，第一个人最开始可以取1个或2个，不能操作时结束。两个人都希望自己取到的数字之和尽量大，并保持绝对理智，求最后他们取到的数字之和之差。$n \leq 4000$。

例题

[Codeforces729F] Financiers Game

标准算法：

动态规划。F[i][j][k][0/1]表示左侧第一个未取的数是第i个，右侧第一个未取的数是第j个，当前k的值和操作方。

这样子的动态规划是O(N^3)的。

例题

[Codeforces729F] Financiers Game

标准算法：

动态规划。F[i][j][k][0/1]表示左侧第一个未取的数是第i个，右侧第一个未取的数是第j个，当前k的值和操作方。

这样子的动态规划是O(N^3)的。

仔细分析，可以发现，k不会超过sqrt(N)级别。复杂度降到了O(N^2.5)。

例题

[Codeforces729F] Financiers Game

标准算法：

动态规划。F[i][j][k][0/1]表示左侧第一个未取的数是第i个，右侧第一个未取的数是第j个，当前k的值和操作方。

这样子的动态规划是O(N^3)的。

仔细分析，可以发现，k不会超过sqrt(N)级别。复杂度降到了O(N^2.5)。

进一步观察可以发现，i与n-j相差不会超过k级别。用这个差代替j。复杂度降到了O(N^2)。

3.4 方程的冗余性

- ✓ 有些方程状态或转移虽多，但是经过分类后发现情况并不多。这时可以重新设立状态，利用状态类别进行转移。

例题

[AHOI2009] 中国象棋

题意：

求N×M的棋盘上有多少种放炮方案，使得它们两两不互相攻击。N, M≤100。

例题

[AHOI2009] 中国象棋

题意：

求N×M的棋盘上有多少种放炮方案，使得它们两两不互相攻击。N, M≤100。

标准算法：

动态规划，f[i][j][k]表示前i行放1个炮的有j列，放2个炮的有k列。

例题

[Codeforces607B] Zuma

题意：

给出一个长度为N的数字串，每次可以从中提取一个回文子串。求至少要提取多少次，才能把整个串取完。N≤500。

例题

[Codeforces607B] Zuma

题意：

给出一个长度为N的数字串，每次可以从中提取一个回文子串。求至少要提取多少次，才能把整个串取完。N≤500。

标准算法：

f[i][j]表示取完区间[i,j]的所有字符需要的最少操作数。

可以讨论c[i]和c[j]是否相等来进行转移。

例题

[Codeforces607B] Zuma

题意：

给出一个长度为N的数字串，每次可以从中提取一个回文子串。求至少要提取多少次，才能把整个串取完。N≤500。

标准算法：

f[i][j]表示取完区间[i,j]的所有字符需要的最少操作数。

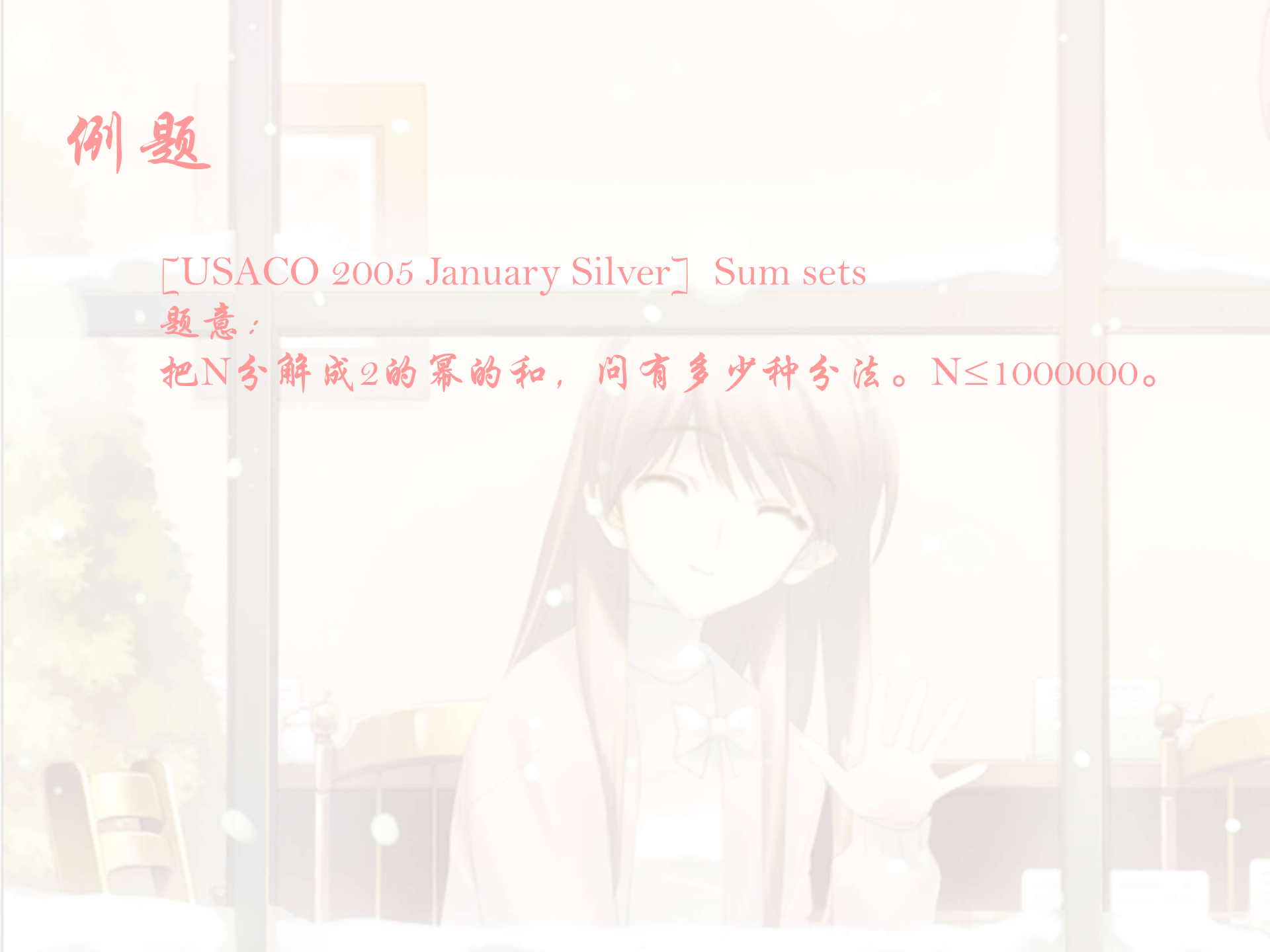
可以讨论c[i]和c[j]是否相等来进行转移。

例题

[USACO 2005 January Silver] Sum sets

题意：

把N分解成2的幂的和，问有多少种分法。N≤1000000。



例题

[USACO 2005 January Silver] Sum sets

题意：

把N分解成2的幂的和，问有多少种分法。N≤1000000。

标准算法：

假设序列中每个幂都已经从小到大排序。

1. 当 n 为奇数时，f[n] = f[n-1]，因为只需在所有的序列前添加一个1即可，所有的序列同时延迟 1 位。

2. 当 n 为偶数时，分为两种情况：

某个序列首位为1，则该序列由 f[n-1]延长而来

某个序列首位为2，则该序列没有1，将该序列的所有元素除以 2，则是f(n/2)的序列。

即f[n] = f[n-1]+f[n/2]。

3.4 方程的冗余性

- ✓ 动态规划时，特别是两种同类型的状态表示，如果可以通过其中多维表示另一维，或是用一些巧妙的方法把两者结合起来，可以达到降低时间、空间、代码复杂度的效果。

例题

[Codeforces570E] Pig and Palindromes

题意：

给出一个N×M的字符矩阵，求从左上角到右下角、每次只能往右或者往下走，经过的路径上所有字符恰好为一个回文串的方案数。N, M≤500。

例题

[Codeforces570E] Pig and Palindromes

题意：

给出一个N×M的字符矩阵，求从左上角到右下角、每次只能往右或者往下走，经过的路径上所有字符恰好为一个回文串的方案数。N, M≤500。

标准算法：

f[i][j][k][l]表示从左上角走到(i,j)，从(k,l)走到右下角，路径上的字符恰好成为回文串前后缀的方案数，每次往四个方向转移。这样的复杂度是O(N^4)的。

例题

[Codeforces570E] Pig and Palindromes

题意：

给出一个N×M的字符矩阵，求从左上角到右下角、每次只能往右或者往下走，经过的路径上所有字符恰好为一个回文串的方案数。N, M≤500。

标准算法：

f[i][j][k][l]表示从左上角走到(i,j)，从(k,l)走到右下角，路径上的字符恰好成为回文串前后缀的方案数，每次往四个方向转移。这样的复杂度是O(N^4)的。

考虑优化。若已经走的步数i确定，记录两个点的横坐标，纵坐标就可以直接推算出来了。f[i][j][k]表示走了i步，从左上角出发走到了第j行，从右下角出发走到了第k行的方案数。这样复杂度就降到了O(N^3)。

基础练习

[CQOI2007] 涂色
[51Nod1201] 整数划分
[洛谷P1373] 小a和uim之大逃离
[Codeforces762D] Maximum path
[Codeforces721E] Road to Home

3.5 预处理技巧

- ✓ 动态规划之前，可以去除一些必定不合法的情况。

3.5 预处理技巧

- ✓ 前缀和差分法是动态规划预处理的重要方法，能够优化区间转移的过程。

例题

[Codeforces498B] Name That Tune

题意：

用T秒时间按顺序听N首歌，第i首歌播放时间为ti秒，且每播放一秒都会有pi的概率被识别出来，跳到下一首。求听歌数量的期望。$1 \le N,T \le 5000$。

例题

[Codeforces498B] Name That Tune

标准算法：

动态规划。f[i][j]表示前i秒时间听了j首歌的概率，则有：

f[i][j]=∑f[i-k][j-1]*(1-pj)^(k-1)*pj。

这个方程直接转移的时间复杂度为O(NT^2)。

例题

[Codeforces498B] Name That Tune

标准算法：

动态规划。f[i][j]表示前i秒时间听了j首歌的概率，则有：f[i][j]=∑f[i-k][j-1]*(1-pj)^(k-1)*pj。

这个方程直接转移的时间复杂度为O(NT^2)。

尝试进行优化。观察可以发现，f[i+1][j]与f[i][j]的方程基本不变，考虑直接从f[i][j]转移到f[i+1][j]。时间复杂度降到O(NT)。

例题

[APIO2009] Oil

题意：

对于一个N×M的矩阵，求出三个边长为K的不重叠子方阵，使得这三个子方阵内部元素和之积最大。N, M≤1500。

例题

[APIO2009] Oil

题意：

对于一个N×M的矩阵，求出三个边长为K的不重叠子方阵，使得这三个子方阵内部元素和之积最大。N, M≤1500。

标准算法：

首先把所有位置边长为K的子方阵内部的元素和预处理出来，之后分类讨论动态规划即可。

3.5 预处理技巧

- ✓ 前驱的记录是另一种重要方法，能够使动态规划快速找到转移的状态。

例题

[BeiJing2010] 取数游戏

题意：

给出一个长度为N数列Ai，要求按顺序取出尽量多的数，并且满足相邻两个取出的数最大公约数大于L。N≤50000，Ai≤1000000。

例题

[BeiJing2010] 取数游戏

题意：

给出一个长度为N数列Ai，要求按顺序取出尽量多的数，并且满足相邻两个取出的数最大公约数大于L。N≤50000，Ai≤1000000。

标准算法：

开桶数组记录每个数前面最近的能够整除某个数字（不小于L）的数的位置，如果当前数的因数中有这个数，那么当前数就可以接在对应数的后面。

例题

[Baltic2008] Elect

题意：

给出一个长度为N的数列Ai，要求从中选出若干个数，使得这些数字之和超过总和的一半，并且去掉选中的任意一个数之后剩下的选中数字之和不超过总和的一半。求最大的选出数字和。$N \leq 300$，$\sum Ai \leq 100000$。

例题

[Baltic2008] Elect

题意：

给出一个长度为N的数列Ai，要求从中选出若干个数，使得这些数字之和超过总和的一半，并且去掉选中的任意一个数之后剩下的选中数字之和不超过总和的一半。求最大的选出数字和。N≤300，∑Ai≤100000。

标准算法：

考虑每个数字选或者不选。f[i][j]表示用前i大的数字能否达到和为j的状态，状态用0/1表示，其中i这一维可以压缩掉。需要对g[]排序，每次用s/2+1..s/2+g[i]更新答案。

例题

[Baltic2008] Elect

题意：

给出一个长度为N的数列Ai，要求从中选出若干个数，使得这些数字之和超过总和的一半，并且去掉选中的任意一个数之后剩下的选中数字之和不超过总和的一半。求最大的选出数字和。N≤300，∑Ai≤100000。

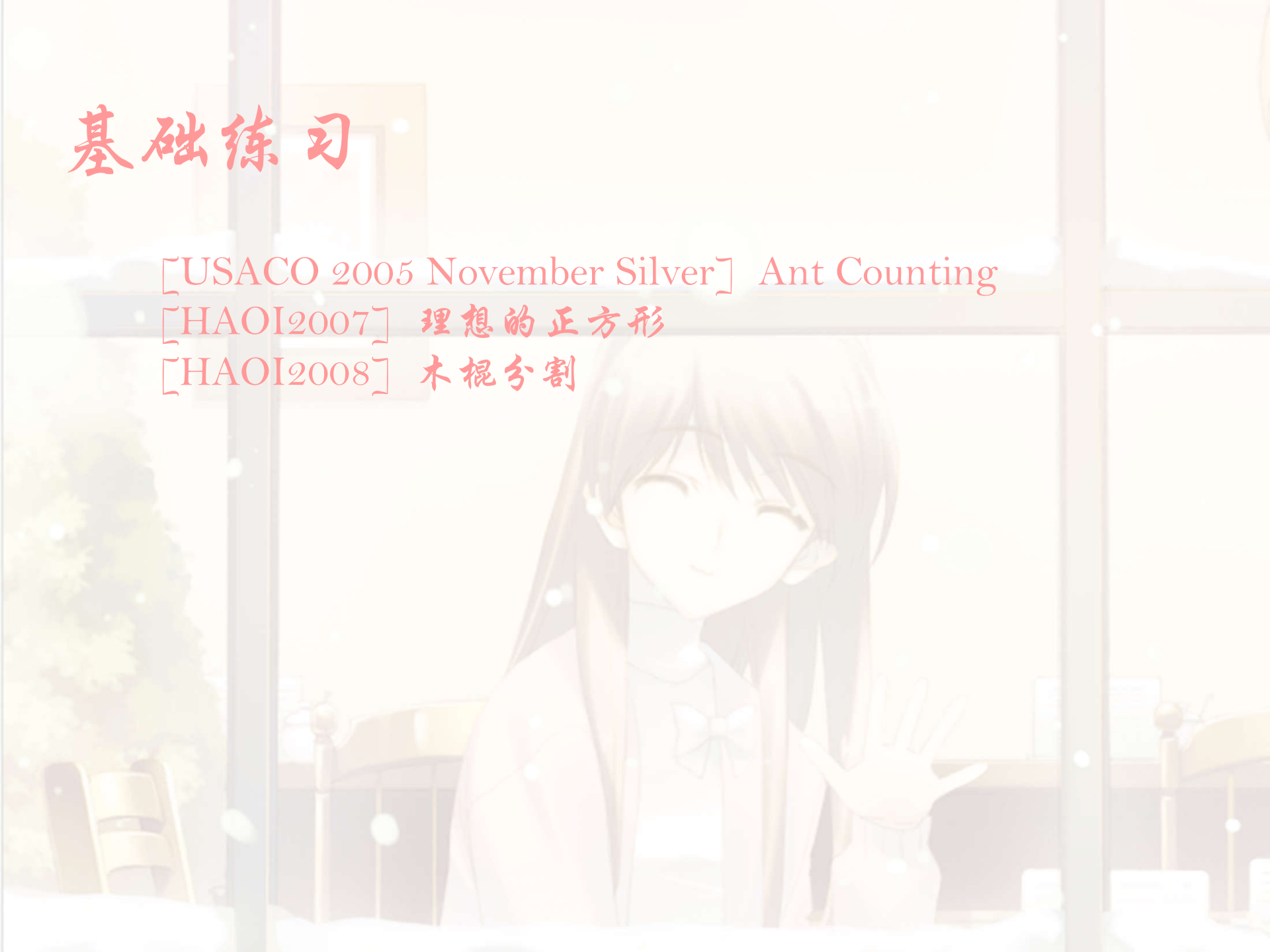
标准算法：

考虑每个数字选或者不选。f[i][j]表示用前i大的数字能否达到和为j的状态，状态用0/1表示，其中i这一维可以压缩掉。需要对g[]排序，每次用s/2+1..s/2+g[i]更新答案。

考虑到当f[j]为1时，之后的任何更新都不会对其产生影响。可以用并查集维护每个状态前最后一个“0”的位置，这样子可以更快地更新状态。

基础练习

[USACO 2005 November Silver] Ant Counting
[HAOI2007] 理想的正方形
[HAOI2008] 木棍分割



3.6 常数优化策略

- ✓ 对于常数优化空间较大的问题，合理地进行剪枝可能会对解题起到决定性作用。
- ✓ 但对于大多数问题，理论复杂度仍然是算法效率的决定性因素。

例题

[COCI2015 Round1] UZASTOPNI

题意：

给出一棵N个点有根树，要求提取出一个与根相连的树形连通块，满足：①点权Vi两两不同；②这棵树的每棵子树对应的点权子集是数轴上一段连续的区间。求满足要求的连通块形成的点权集合总数。N≤10000，Vi≤100。

例题

[COCI2015 Round1] UZASTOPNI

题意：

给出一棵N个点有根树，要求提取出一个与根相连的树形连通块，满足：①点权Vi两两不同；②这棵树的每棵子树对应的点权子集是数轴上一段连续的区间。求满足要求的连通块形成的点权集合总数。N≤10000，Vi≤100。

标准算法：

f[i][j][k]表示第i个点，排好序后的数值区间为[j,k]，这样是否可以实现。

f[i][j][k]必定可以由f[Ei][x][y]转移过来。粗略地分析，这样的时间复杂度大概是O(NV^3)。这样子的时间和空间复杂度都不能承受。

例题

[COCI2015 Round1] UZASTOPNI

题意：

给出一棵N个点有根树，要求提取出一个与根相连的树形连通块，满足：①点权Vi两两不同；②这棵树的每棵子树对应的点权子集是数轴上一段连续的区间。求满足要求的连通块形成的点权集合总数。N≤10000，Vi≤100。

标准算法：

但是，注意到其中x、y不能包含任何Ei的祖先的权值。因此，随着递归层数的不断深入，权值范围必定不断缩小。并且，题目要求点权两两不同，这也就意味着递归层数必定不会超过V层。

例题

[COCI2015 Round1] UZASTOPNI

题意：

给出一棵N个点有根树，要求提取出一个与根相连的树形连通块，满足：①点权Vi两两不同；②这棵树的每棵子树对应的点权子集是数轴上一段连续的区间。求满足要求的连通块形成的点权集合总数。N≤10000，Vi≤100。

标准算法：

进行预处理。预处理出b[i]——i到根的路径上是否有与i点权相同的点，l[i]、r[i]——i的子树可能的权值区间的左右界。这些预处理的结果，能够大大提升动态规划效率。再加上一些剪枝（比如求乘积，那么枚举的一个数等于零，就不用第二重枚举另一个数了），程序的运行效率会更高。

例题

[COCI2015 Round1] UZASTOPNI

题意：

给出一棵N个点有根树，要求提取出一个与根相连的树形连通块，满足：①点权Vi两两不同；②这棵树的每棵子树对应的点权子集是数轴上一段连续的区间。求满足要求的连通块形成的点权集合总数。N≤10000，Vi≤100。

标准算法：

进行预处理。预处理出b[i]——i到根的路径上是否有与i点权相同的点，l[i]、r[i]——i的子树可能的权值区间的左右界。这些预处理的结果，能够大大提升动态规划效率。再加上一些剪枝（比如求乘积，那么枚举的一个数等于零，就不用第二重枚举另一个数了），程序的运行效率会更高。然后就能把这题0.00s过了？