第十次直播课

习题讲解

李嘉政

Dec 2023

Table of Contents

Table of Contents

Solution

做法太多了，最简单的做法当然是直接背包 dp，设 $f_{i,j}$ 为前 i 个街道花了 j 元的方案数，转移就是简单的 $f_{i,j}=\sum_{l=1}^{m} f_{i-1,j-l}$。时间复杂度 $\mathcal{O}(nkm)$，前缀和可以优化到 $\mathcal{O}(nk)$。
当然实际上存在直接隔板法容斥的做法，时间复杂度 $\mathcal{O}(n+m+k)$，更进一步可以用生成函数，时间复杂度改成 $\mathcal{O}(n\log k+m\log m\log k)$。

Table of Contents

Solution

f_i 表示从 0 走到 i 的方案数，转移显然是 $f_i = f_{i-1} + f_{i-2}$，那些坏掉的位置的 f 不转移，设置为 0 即可。时间复杂度 $\mathcal{O}(n+m)$。

Table of Contents

Solution

枚举第 i 个人作为最高的那个人，此时等价于要求以它为结尾的最长不降序列和以它为开头的最长不升序列。用 f_i 表示前者，转移是显然的 $f_i = \max_{a_j \le a_i, j<i} f_j + 1$，后者求法类似。暴力是 $\mathcal{O}(n^2)$，树状数组可以做到 $\mathcal{O}(n \log n)$。

Table of Contents

Solution

$f_{i,j}$ 表示已经构造了前 i 个数，第 i 个数是 j 的方案数。转移就是 $f_{i,j} = \sum_{l|j} f_{i-1,l}$，这里的时间复杂度是经典的调和级数，时间复杂度 $\mathcal{O}(nk \log k)$。

Table of Contents

Solution

简单的背包，$f_{i,j}$ 表示前 i 个数是否可以组成 j，转移就是 $f_{i,j} = f_{i-1,j} \vee f_{i-1,j-a_i}$。最后枚举机器一用了多少时间即可。时间复杂度 $\mathcal{O}(n^2 t)$，bitset 可以优化到 $\mathcal{O}(\frac{n^2 t}{w})$。

Table of Contents

Solution

考虑 f_i 表示前 $1 \sim i$ 位置的油桶都已经确定，且 i 这个位置放一个油桶的方案数。转移显然是 $f_i = \sum_{j=0}^{i-k-1} f_j$。直接做时间复杂度不优秀，前缀和优化即可。时间复杂度 $\mathcal{O}(n)$。

Table of Contents

Solution

$f_{i,j,k,0/1}$ 表示从 $(0,0)$ 走到了 (i,j)，用了 k 次转向，当前方向是下/右的方案数。转移就是直接枚举当前是否转向即可。时间复杂度 $\mathcal{O}(nmK)$，容易证明答案不会超过 long long。

Table of Contents

Solution

f_T 到时刻 T 时的最大利润，我们认为电影放完才有利润。转移显然是枚举放第 i 部电影，于是 $f_{T+T_i+K} \leftarrow \max(f_{T+T_i+K}, f_T + P_i)$。注意最后一部电影放完后不需要再等 K 时间，提前取 $\max$ 即可。时间复杂度 $\mathcal{O}(nm)$。

Table of Contents

Solution

求 LIS 可以做到 $\mathcal{O}(n \log n)$。考虑 f_i 表示以 i 为结尾的最长上升子序列长度，转移显然是 $f_i = \max_{j<i, a_j<a_i} f_j + 1$。如何优化这个转移呢？由于 dp 时是从前往后推的，所以等价于我们需要知道所有满足 $a_j < a_i$ 中最大的 f 值。

有两个思路可以解决这个问题。

思路一是，我们希望从这个结构中获取一些信息，可以观察到一个重要性质：如果某个 $a_i \le a_j$ 且 $f_i \ge f_j$ 时，对所有大于 i 的位置，与其选择 j 一定不如选择 i。这意味着什么呢？在取不劣的情况下，能够被选择的位置如果按 f 从小到大排序后，则 a 也会是升序的。有了这个思路，我们想着维护一个数组 g，其中 g_i 表示当前情况下，$f = i$ 的位置中 a 最小的。此时 g 数组是单调升序的。在求 f 时，可以直接在 g 中二分出第一个小于当前 a 的位置，然后 f 就是在 g 中的位置加一。而在求出当前的 f，再反过来更新 g 即可。时间复杂度 $\mathcal{O}(n \log n)$。

Solution

第二种思路是不再去结构所拥有的一些性质了。我们可以直接将所有满足 $a_j < a_i$ 中最大的 f 值这个信息用其他结构快速维护，比如值域树状数组 (离散化) 等。时间复杂度也是 $\mathcal{O}(n \log n)$。

Table of Contents

Solution

同上题。

Table of Contents

Solution

$f_{i,j}$ 表示序列一以 i 为结尾，序列二以 j 为结尾的最长公共子序列的长度。转移显然是如果 $a_i \neq b_j$，则 $f_{i,j} = 0$；否则 $f_{i,j} = \max_{k<i,l<j} f_{k,l} + 1$。直接转移时间复杂度肯定不优，考虑前缀和优化 (二维前缀和)，时间复杂度优化到 $\mathcal{O}(nm)$。