贪心和dp

吴思扬

2019.3.

贪心

贪心可以为解决最优化问题提供策略。

贪心

例

有 n 个盒子，每个盒子有重量 w_i 和强度 s_i。
你要选则尽可能多的盒子堆成一竖列，使得每个盒子的强度不小于之上的盒子的重量和。

贪心

要确定最多能选多少个盒子，首先要确定选出的盒子排列顺序。

贪心

要确定最多能选多少个盒子，首先要确定选出的盒子排列顺序。考虑两个盒子 i 和 j 的顺序。如果 $s_j \geq w_i$ 且 $w_j > s_i$ 则 i 必须在 j 上面。对于相邻两个盒子都要满足这个条件，因此可以对所有盒子按照 $s_i + w_i$从小到大排序。

贪心

要确定最多能选多少个盒子，首先要确定选出的盒子排列顺序。考虑两个盒子 i 和 j 的顺序。如果 $s_j \geq w_i$ 且 $w_j > s_i$ 则 i 必须在 j 上面。对于相邻两个盒子都要满足这个条件，因此可以对所有盒子按照 $s_i + w_i$从小到大排序。
之后可以进行一次 dp，令 $dp(i,\ j)$ 表示已经考虑了前 i 个盒子，选了 j 个，w 和的最小值是多少。

Balanced Sequence

有 n 个括号序列，你可以把它们排列之后首尾相连拼在一起，然后删去一些字符使得剩下的是合法括号序列。
$\sum |S| \leq 5 * 10^6$

Balanced Sequence

首先可以对每个串把匹配的括号删去，剩下来为 a 个) 和 b 个 (。
令 sum 表示和，m 为前缀和最小值，则能保留的最长长度为 $n - sum + 2m$。由于 n 固定，则要最大化 m。
$a < b$ 肯定排在 $a > b$ 前面。
同为 $a < b$ 则 a 小的排前面。
同为 $a > b$ 则 b 大的排前面。

打怪兽

打一个怪兽要先掉 a_i 滴血再补 b_i 滴血。树上每个点有个怪兽，打完一个点父亲才能打这个点，问最少要多少血使得血始终非负。

$n \leq 10^6$

打怪兽

如果可以任意排列则方案与上一题一样。
每次先选择顺序最小的，如果其父亲是根，则打这个怪兽，否则将其和父亲合并，意思是打完父亲就打它。(a_i, b_i)，(a_j, b_j) 合并完为
$(max(a_i, a_i - b_i + a_j), -a_i + b_i - a_j + b_j + max(a_i, a_i - b_i + a_j))$。

树形dp

树形dp用于解决树上问题。

树形dp

树形dp用于解决树上问题。
状态常常用 $f(u, S)$ 来表示已经对子树 u 这个子问题进行过求解，状态为 S 的值。

树形dp

树形dp用于解决树上问题。
状态常常用 $f(u, S)$ 来表示已经对子树 u 这个子问题进行过求解，状态为 S 的值。
转移往往会有子树合并，以及从 u 转移到 u 的父亲 par_u。

树形dp

其代码常常如下所示：

```
dfs(u)
    initial f(u)
    foreach v in son[u]
        dfs(v)
        update f(u) with f(v)
```

树形dp

例

有一颗 n 个节点 $n-1$ 条边的无向树，每条边 (u, v) 有边权 w。
问树上最远两个点的距离

树形dp

例

有一颗 n 个节点 $n-1$ 条边的无向树，每条边 (u, v) 有边权 w。
问树上最远两个点的距离
数据范围：$1 \leq n \leq 10^5$, $|w| \leq 10^9$。

树形dp

令 $dp(u)$ 表示对于以 u 为根的子树，u 子树中距离 u 距离最远的点的距离是多少。

树形dp

令 $dp(u)$ 表示对于以 u 为根的子树，u 子树中距离 u 距离最远的点的距离是多少。
用 $dp(v)$ 的值去更新 v 的父亲 u 的值时只用做 $dp(u) = \max\{dp(u), dp(v) + w(u, v)\}$。

树形dp

例

有一颗 n 个节点 $n-1$ 条边的无向树，树上节点用 $1, 2, \ldots, n$ 编号。
初始时每个节点都时白色。如果选中了节点 u，则对于树中的每条边 (u, v)，v 都会被染成黑色。注意 u 自身不会被染黑。
现在总共要选择恰好 k 个点，问有多少种方案使得所有节点被染黑。

树形dp

例

有一颗 n 个节点 $n-1$ 条边的无向树，树上节点用 $1, 2, \ldots, n$ 编号。
初始时每个节点都时白色。如果选中了节点 u，则对于树中的每条边 (u, v)，v 都会被染成黑色。注意 u 自身不会被染黑。
现在总共要选择恰好 k 个点，问有多少种方案使得所有节点被染黑。
数据范围：$1 \leq n \leq 10^5$, $1 \leq k \leq \min(n, 100)$。

树形dp

令 $dp(u,\ k,\ color,\ choice)$ 表示对于以 u 为根的子树，里面选了 k 个点，u 的颜色为黑/白，u 有没有被选中。u 子树外点的选择情况之能影响到 u 子树中 u 点的颜色，u 子树中只有 u 点的选择情况能影响到 u 子树外的点的颜色。

树形dp

令 $dp(u,\ k,\ color,\ choice)$ 表示对于以 u 为根的子树，里面选了 k 个点，u 的颜色为黑/白，u 有没有被选中。u 子树外点的选择情况之能影响到 u 子树中 u 点的颜色，u 子树中只有 u 点的选择情况能影响到 u 子树外的点的颜色。
时间复杂度：$O(nk^2)$。

树形dp

时间复杂度：$O(nk^2)$？

树形dp

时间复杂度：$O(nk^2)$？
对合并的两棵子树大小分类讨论，可以证明是 $O(nk)$。

数位dp

如果题目给定了一些非常大的数字，问你有多少种符合一系列与给定的大数有关的条件的数字，数位dp就可以帮助解决这类问题。

数位dp

如果题目给定了一些非常大的数字，问你有多少种符合一系列与给定的大数有关的条件的数字，数位dp就可以帮助解决这类问题。
可以将数位dp想像成逐位填数字的方法。dp状态为 $f(i, S)$，表示已经填好了前/后 i 个数字，题目中的条件满足的情况为 S 的方案数。

数位dp

如果题目给定了一些非常大的数字，问你有多少种符合一系列与给定的大数有关的条件的数字，数位dp就可以帮助解决这类问题。

可以将数位dp想像成逐位填数字的方法。dp状态为 $f(i, S)$，表示已经填好了前/后 i 个数字，题目中的条件满足的情况为 S 的方案数。

转移时枚举第 $i + 1$ 位数字填的是什么，进行转移。可以发现转移时间复杂度还跟进制相关。实践可以发现，有时对 2 进制或 3 进制进行数位dp较为高效。

数位dp

例

给 n, m, k，求有多少 $0 \leq i \leq n, 0 \leq j \leq min(i, m)$ 满足 $k | \binom{i}{j}$。
数据范围：$1 \leq n, m \leq 10^{18}, 1 \leq k \leq 100$，$k$为质数。

数位dp

lucas定理：$\binom{ak+b}{ck+d} = \binom{a}{c} \cdot \binom{b}{d} (\mathrm{mod}\, k),\ 0 \le b, d < k$。

数位dp

lucas定理：$\binom{ak+b}{ck+d} = \binom{a}{c} \cdot \binom{b}{d} (\mathrm{mod}\, k),\ 0 \le b, d < k$。
令 $dp(x,\ f,\ a,\ b,\ c)$ 表示填了前 x 位，模 k 是否为 0，i 是否 $\le n$，j 是否 $\le m$，j 是否 $\le i$。

数位dp

lucas定理：$\binom{ak+b}{ck+d} = \binom{a}{c} \cdot \binom{b}{d} (\mathrm{mod}\, k),\ 0 \le b, d < k$。
令 $dp(x,\ f,\ a,\ b,\ c)$ 表示填了前 x 位，模 k 是否为 0，i 是否 $\le n$，j 是否 $\le m$，j 是否 $\le i$。
转移直接枚举下一位的两个数即可。

数位dp

例

定义 $S(n)$ 为将 n 在 10 进制下的所有数位从小到大排序后得到的数。例如：$S(1) = 1, S(50394) = 3459, S(323) = 233$。给定 X 求 $\sum_{i=1}^{X} S(i)$ 对 $10^9 + 7$ 取模的结果。

数位dp

例

定义 $S(n)$ 为将 n 在 10 进制下的所有数位从小到大排序后得到的数。例如：$S(1) = 1, S(50394) = 3459, S(323) = 233$。
给定 X 求 $\sum_{i=1}^{X} S(i)$ 对 $10^9 + 7$ 取模的结果。
数据范围：$1 \leq X \leq 10^{700}$。

数位dp

考虑如何计算答案，可以通过分别计算每一位对总和的贡献来求。定义 $cnt(i,\ x)$ 为 $S(1),\ S(2),\ ...,\ S(X)$ 中第 i 位为 x 的数量。

数位dp

考虑如何计算答案，可以通过分别计算每一位对总和的贡献来求。定义 $cnt(i,\ x)$ 为 $S(1),\ S(2),\ ...,\ S(X)$ 中第 i 位为 x 的数量。
直接求 $cnt(i,\ x)$ 仍然较为困难，考虑差分。令 $dlt(i,\ x)$ 为第 i 位上有多少个数 $\geq\ x$。对于一个 $S(y)$ 中第 i 位 $\geq\ x$ 的数 y，可以发现其必须满足有至少 i 个数位 $\geq\ x$，这样就可以进行dp了。

数位dp

考虑如何计算答案，可以通过分别计算每一位对总和的贡献来求。定义 $cnt(i,\ x)$ 为 $S(1),\ S(2),\ ...,\ S(X)$ 中第 i 位为 x 的数量。

直接求 $cnt(i,\ x)$ 仍然较为困难，考虑差分。令 $dlt(i,\ x)$ 为第 i 位上有多少个数 $\geq\ x$。对于一个 $S(y)$ 中第 i 位 $\geq\ x$ 的数 y，可以发现其必须满足有至少 i 个数位 $\geq\ x$，这样就可以进行dp了。

另 $dp(i,\ j,\ x,\ cmp)$ 表示填了前 i 位，有至少 j 个数字 $\geq\ x$，与 N 的大小关系位 cmp。

数位dp

考虑如何计算答案，可以通过分别计算每一位对总和的贡献来求。定义 $cnt(i,\ x)$ 为 $S(1),\ S(2),\ ...,\ S(X)$ 中第 i 位为 x 的数量。

直接求 $cnt(i,\ x)$ 仍然较为困难，考虑差分。令 $dlt(i,\ x)$ 为第 i 位上有多少个数 $\geq\ x$。对于一个 $S(y)$ 中第 i 位 $\geq\ x$ 的数 y，可以发现其必须满足有至少 i 个数位 $\geq\ x$，这样就可以进行dp了。

另 $dp(i,\ j,\ x,\ cmp)$ 表示填了前 i 位，有至少 j 个数字 $\geq\ x$，与 N 的大小关系位 cmp。

转移时直接枚举第 $i\ +\ 1$ 位数字是多少即可。

状压dp

状压dp可以用来解决状态较为复杂的问题，时间复杂度常常是指数级别。

状压dp

状压dp可以用来解决状态较为复杂的问题，时间复杂度常常是指数级别。
dp的状态为 $f(i,\ S)$。状压dp将所要考虑的所有状态一一对应到一个实数值 i 上，这样实现起来较为简单。最常见的例子就是对于一个集合 $S\ =\ \{0,\ 1,\ ...,\ n\ -\ 1\}$ 的一个子集 $A\ =\ \{a_0,\ a_1,\ ...,\ a_k\}$，与数 $\sum_{i\ =\ 0}^{k}\ a_i\ 2^i$ 一一对应。

状压dp

状压dp可以用来解决状态较为复杂的问题，时间复杂度常常是指数级别。
dp的状态为 $f(i,\ S)$。状压dp将所要考虑的所有状态一一对应到一个实数值 i 上，这样实现起来较为简单。最常见的例子就是对于一个集合 $S\ =\ \{0,\ 1,\ ...,\ n\ -\ 1\}$ 的一个子集 $A\ =\ \{a_0,\ a_1,\ ...,\ a_k\}$，与数 $\sum_{i\ =\ 0}^{k}\ a_i\ 2^i$ 一一对应。
转移方程直接枚举下一个规模更小的子问题即可。

状压dp

例

你准备去 N 个国家进行旅行，去第 i 个国家的旅行会在第 s_i 天的早上出发，第 $s_i + len_i - 1$ 天的晚上回家。

状压dp

例

你准备去 N 个国家进行旅行，去第 i 个国家的旅行会在第 s_i 天的早上出发，第 $s_i + len_i - 1$ 天的晚上回家。
你有 P 本护照，在每次旅行前必须让其中一本护照办理该国的签证，如果在第 x 天开始对某本护照办理第 i 个国家的签证，那么第 x 天不能在旅行，且第 $x + t_i$ 天中午可完成签证拿回该护照（允许在旅行时拿到）。判断旅行计划能否完成，如果能，给出一种签证方案（时间及哪本护照）。

状压dp

例

你准备去 N 个国家进行旅行，去第 i 个国家的旅行会在第 s_i 天的早上出发，第 $s_i + len_i - 1$ 天的晚上回家。
你有 P 本护照，在每次旅行前必须让其中一本护照办理该国的签证，如果在第 x 天开始对某本护照办理第 i 个国家的签证，那么第 x 天不能在旅行，且第 $x + t_i$ 天中午可完成签证拿回该护照（允许在旅行时拿到）。判断旅行计划能否完成，如果能，给出一种签证方案（时间及哪本护照）。
数据范围：$1 \leq N \leq 22, 1 \leq P \leq 2$。

状压dp

$P = 1$ 时怎么做？

状压dp

$P = 1$ 时怎么做？
由于 N 的范围只有 22，不妨考虑状压dp。令 $f(S)$ 表示办完 S 集合的所有签证，拿回护照的时间最早是多少。

状压dp

$P = 1$ 时怎么做？
由于 N 的范围只有 22，不妨考虑状压dp。令 $f(S)$ 表示办完 S 集合的所有签证，拿回护照的时间最早是多少。
转移时直接枚举下一个办签证的国家即可。

状压dp

可以发现，如果有 2 个护照，对每个护照办签证的时间基本上是相似的，差别仅仅在于每个护照所签的国家进行旅行时该护照必须在身上。

状压dp

可以发现，如果有 2 个护照，对每个护照办签证的时间基本上是相似的，差别仅仅在于每个护照所签的国家进行旅行时该护照必须在身上。
不妨修改 $f(S)$ 的定义，表示用一个护照办完 S 集合的所有签证，并且用这个护照旅行 S 的所有国家的拿回护照的最早时间。

状压dp

可以发现，如果有 2 个护照，对每个护照办签证的时间基本上是相似的，差别仅仅在于每个护照所签的国家进行旅行时该护照必须在身上。
不妨修改 $f(S)$ 的定义，表示用一个护照办完 S 集合的所有签证，并且用这个护照旅行 S 的所有国家的拿回护照的最早时间。

转移时枚举下一个办签证的国家，判断是否可行并更新其它的 f 值。在判断可行时，可以按照下一个办签证国家时间长短的顺序依次枚举。可以证明这样求出的更新其它 f 的值是单调不减的，所以用指针维护即可。时间复杂度 $O(2^N\ N)$。

状压dp

可以发现，如果有 2 个护照，对每个护照办签证的时间基本上是相似的，差别仅仅在于每个护照所签的国家进行旅行时该护照必须在身上。
不妨修改 $f(S)$ 的定义，表示用一个护照办完 S 集合的所有签证，并且用这个护照旅行 S 的所有国家的拿回护照的最早时间。

转移时枚举下一个办签证的国家，判断是否可行并更新其它的 f 值。在判断可行时，可以按照下一个办签证国家时间长短的顺序依次枚举。可以证明这样求出的更新其它 f 的值是单调不减的，所以用指针维护即可。时间复杂度 $O(2^N\ N)$。
最后求答案是只需要枚举第一个护照办理的签证即可。

NOIP2017 宝藏

参与考古挖掘的小明得到了一份藏宝图，藏宝图上标出了 n 个深埋在地下的宝藏屋，也给出了这 n 个宝藏屋之间可供开发的 m 条道路和它们的长度。

小明决心亲自前往挖掘所有宝藏屋中的宝藏。但是，每个宝藏屋距离地面都很远，也就是说，从地面打通一条到某个宝藏屋的道路是很困难的，而开发宝藏屋之间的道路则相对容易很多。

小明的决心感动了考古挖掘的赞助商，赞助商决定免费赞助他打通一条从地面到某个宝藏屋的通道，通往哪个宝藏屋则由小明来决定。

在此基础上，小明还需要考虑如何开凿宝藏屋之间的道路。已经开凿出的道路可以任意通行不消耗代价。每开凿出一条新道路，小明就会与考古队一起挖掘出由该条道路所能到达的宝藏屋的宝藏。另外，小明不想开发无用道路，即两个已经被挖掘过的宝藏屋之间的道路无需再开发。

NOIP2017 宝藏

新开发一条道路的代价是：
这条道路的长度 x 从赞助商帮你打通的宝藏屋到这条道路起点的宝藏屋所经过的宝藏屋的数量（包括赞助商帮你打通的宝藏屋和这条道路起点的宝藏屋）。
请你编写程序为小明选定由赞助商打通的宝藏屋和之后开凿的道路，使得工程总代价最小，并输出这个最小值。

NOIP2017 宝藏

令 $f(S,\ i)$ 表示 S 集合内的点已经联通，最深一层深度为 i。

NOIP2017 宝藏

令 $f(S,\ i)$ 表示 S 集合内的点已经联通，最深一层深度为 i。转移时考虑枚举 $T \subseteq S^c$，用 $f(S,\ i) + cost(S, T) \times i$ 更新 $f(S \cup T,\ i + 1)$。其中 $cost(S,\ T)$ 为 T 集合中的每个点 u 找到 S 中的一个点 v 使得边长最小的总和。

NOIP2017 宝藏

令 $f(S,\ i)$ 表示 S 集合内的点已经联通，最深一层深度为 i。
转移时考虑枚举 $T \subseteq S^c$，用 $f(S,\ i) + cost(S, T) \times i$ 更新 $f(S \cup T,\ i + 1)$。其中 $cost(S,\ T)$ 为 T 集合中的每个点 u 找到 S 中的一个点 v 使得边长最小的总和。
考虑这样更新的正确性。如果 T 中 u 找到的 v 均在 i 层则符合定义。如果不在 i 层，只会算大答案，说明存在更优的解，并不会影响正确性。

ARC081F

$w \times h$ 的矩阵，每个格子为黑色或者白色。你可以进行任意次操作，每次操作将某一行或者某一列颜色取反。问可以得到的最大全黑子矩形的面积。

ARC081F

$w \times h$ 的矩阵，每个格子为黑色或者白色。你可以进行任意次操作，每次操作将某一行或者某一列颜色取反。问可以得到的最大全黑子矩形的面积。
数据范围：$2 \leq w, h \leq 2000$。

ARC081F

经过观察可以发现对于一个 2×2 的矩形，若黑色个数为奇数则无论怎么变化都不能全黑，否则可以全黑。

ARC081F

经过观察可以发现对于一个 2×2 的矩形，若黑色个数为奇数则无论怎么变化都不能全黑，否则可以全黑。
令 $a_{i,j}$ 表示 $(i, j), (i, j+1), (i+1, j), (i+1, j+1)$ 的黑色个数的就行。问题转化为最大的全 0 矩形。

ARC081F

经过观察可以发现对于一个 2×2 的矩形，若黑色个数为奇数则无论怎么变化都不能全黑，否则可以全黑。
令 $a_{i,j}$ 表示 $(i, j), (i, j+1), (i+1, j), (i+1, j+1)$ 的黑色个数的就行。问题转化为最大的全 0 矩形。
从左到右一列一列扫过去，令 f_i 表示第 i 行从当前列开始最多有多少个连续的格子为 0。问题变成求 $\max\{\min\{f_l, \ldots, f_r\} \times (r - l + 1)\}$。

ARC081F

经过观察可以发现对于一个 2×2 的矩形，若黑色个数为奇数则无论怎么变化都不能全黑，否则可以全黑。
令 $a_{i,j}$ 表示 $(i, j), (i, j+1), (i+1, j), (i+1, j+1)$ 的黑色个数的就行。问题转化为最大的全 0 矩形。
从左到右一列一列扫过去，令 f_i 表示第 i 行从当前列开始最多有多少个连续的格子为 0。问题变成求 $\max\{\min\{f_l, \ldots, f_r\} \times (r - l + 1)\}$。
枚举 $min\{f_l, \ldots, f_r\}$。设在 f_i 时取到，则只要求出 l, r 即可。直接扫两遍即可。

CEOI2016 Kangaroo

求有多少个排列 $\{p_1, p_2, \ldots, p_n\}$ 满足 $p_1 = s$, $p_n = t$ 且对于 $1 < i < n$ 满足 $p_{i-1} < p_i$ 且 $p_{i+1} < p_i$ 或 $p_{i-1} > p_i$ 且 $p_{i+1} > p_i$。输出答案对 $10^9 + 7$ 取模。

CEOI2016 Kangaroo

求有多少个排列 $\{p_1, p_2, \ldots, p_n\}$ 满足 $p_1 = s$, $p_n = t$ 且对于 $1 < i < n$ 满足 $p_{i-1} < p_i$ 且 $p_{i+1} < p_i$ 或 $p_{i-1} > p_i$ 且 $p_{i+1} > p_i$。输出答案对 $10^9 + 7$ 取模。
数据范围：$1 \leq s, t \leq n \leq 2000$, $s \neq t$。

CEOI2016 Kangaroo

令 $f(i,\ j,\ x)$ 表示前 i 个点已经填好了，构成了 j 条不包含 s，t 的链，s 和 t 所在的链状态为 x。

CEOI2016 Kangaroo

令 $f(i,\ j,\ x)$ 表示前 i 个点已经填好了，构成了 j 条不包含 $s,\ t$ 的链，s 和 t 所在的链状态为 x。
转移时枚举第 $i\ +\ 1$ 个点连上哪些链即可。

找子矩阵

给定 $n * m$ 个数字，可以修改不超过 A 个数字为 0。问选 B 个不相交的列数为 m 的矩形所能得到的和的最大值是多少。
$1 \leq n \leq 100,\ 1 \leq m \leq 3000,\ 0 \leq A \leq 10000,\ 1 \leq B \leq 3,\ |w_{i,j}| \leq 10^9$

找子矩阵

令 $F(i,\ j,\ k,\ 0/1)$ 表示考虑过前 i 行，改了 j 个数字，k 个矩形开始选了，当前行选不选的最大值。
有转移 $f(i,\ j,\ k,\ 0)\ =\ max(f(i\ -\ 1,\ j,\ k,\ 0),\ f(i\ -\ 1,\ j,\ k,\ 1))$, $f(i,\ j,\ k,\ 1)\ =\ max\{f(i\ -\ 1,\ j\ -\ a,\ k\ -\ 1,\ 0)\ +\ g(i,\ a),\ f(i\ -\ 1,\ j\ -\ a,\ k,\ 1)\ +\ g(i,\ a)\}$。$g(i,\ a)$ 为第 i 行改 a 个数字的和。考虑 $f(i,\ j,\ k,\ 1)$ 的转移，由于 $g(i,\ a\ +\ 1)\ -\ g(i,\ a)\ \geq\ g(i,\ a\ +\ 2)\ -\ g(i,\ a\ +\ 1)$，所以转移具有单调性，直接分治转移即可。

祝大家省选取得好成绩!