LEBOXES 解题报告

1　题目大意

有 n 个盒子，对于第 i 个盒子你有 $\frac{P_i}{100}$ 的概率获得 V_i 美元的钱，而有 $\frac{100-P_i}{100}$ 的概率获得一个钻石。有 m 个物品，第 j 个物品需要花费恰好 C_j 美元的钱和 D_j 个钻石。如果有一定量的钱和钻石，你必定会买尽可能多的物品，每个物品只能购买一次，求你期望获得的物品数。

2　数据范围

$2 \le n \le 30$, $1 \le m \le 30$, $1 \le V_i, C_j \le 10^7$, $0 \le D_j \le 30$, $0 \le P_j \le 100$

3　题目解答

我们最直观的写法就是 2^n 暴力枚举每个盒子的情况，然后用 DP 算出此时可以获得的最多物品数。那么对于这题 $2 \le n \le 30$ 的数据范围，我们很容易想到 meet-in-the-middle 算法。

首先，我们求出 cost[count][d_count]表示：购买恰好 count 个物品，只能使用不超过 d_count 个钻石，最少需要的钱的数量。这个很容易用 DP 在 O(nm^2)的时间内求出：枚举每个物品 k，则 cost[i][j]=min(cost[i][j],cost[i-1][j-d[k]]+c[k])。

现在我们把 n 个盒子分成大小分别为 an 和 bn 的两部分，当然 an＋bn＝n。对于后一部分的 bn 个盒子，我们做一些预处理，用 sta[d_count]存储所有状态中，获得钻石数恰好为 d_count 的状态(money, pro)，其中 money 表示获得的钱数，pro 表示该状态出现的概率，然后我们把这些状态以 money 为关键字排序。

接着我们暴力枚举前一部分 an 个盒子的所有状态，设当前状态获得的钱数和钻石数分别为 a_money 与 a_dia，该状态出现的概率为 a_pro。然后我们枚举在后一部分盒子所获得的钻石数 b_dia，那么如果我们最终想获得 count 个物品，在后一部分盒子中所获得钱的数量 b_money 必定满足：

cost[count][a_dia＋b_dia]≤a_money＋b_money＜cost[count+1][a_dia＋b_dia]

因为如果获得的钱数过少，我们就无法购买 count 个物品，而如果获得的钱数过多，我们就会购买多于 count 个物品。

显然，满足上述条件的所有状态在有序数组 sta[b_dia] 中是一个连续的区间，所以我们只要二分出这个区间，求出这个区间内所有状态的概率之和 b_pro_sum 就行了，此时对答案的贡献即为 count×a_pro×b_pro_sum。

4 复杂度分析

时间复杂度 $O(2^{bn} \cdot \log_2 2^{bn} + 2^{an} \cdot (bn) \cdot m \cdot \log_2 2^{bn})$，即 $O(bn \cdot 2^{bn} + m \cdot (bn)^2 \cdot 2^{an})$，其中 $an + bn = n$。我们取 $an = \dfrac{n}{3}, bn = \dfrac{2n}{3}$ 即可通过此题。

TICKETS 解题报告

1 题目大意

有 n 道菜肴和 m 位顾客，每位顾客有两道喜爱的菜，每道菜只能供应给一位顾客。现在问你最多能招待多少位顾客，使得无论来的是哪些顾客，所有顾客都能够吃到至少一道他喜欢的菜。

2 数据范围

$2 \le n \le 200$, $0 \le m \le 500$

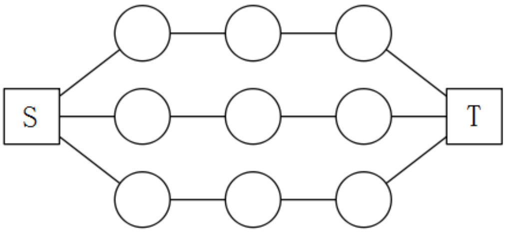
3 题目解答

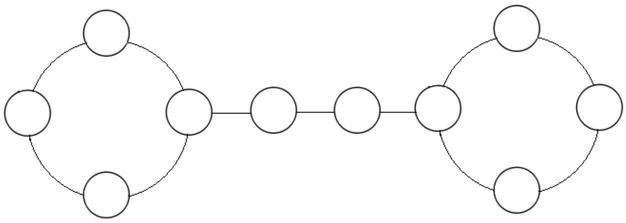
我们建出一个 n 个点 m 条边的无向图，其中每个顶点代表一道菜肴，而每一条边则代表一位顾客，连接着他所喜爱的两道菜所对应的顶点。那么厨师能够卖出 K 张门票的充要条件是：对于无向图中任意大小为 K 的边集 E，与其邻接的点集 V 必定满足 $K = |E| \le |V|$（我们假设边集连通，因为不同连通块之间是互不影响的，可以分开考虑）。

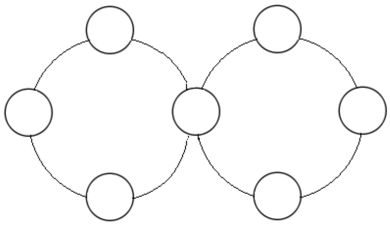
我们构造性地来证明上述结论。首先对于任意一个连通图，必然有 $|E| \ge |V| - 1$。分情况讨论：当 $|E| > |V|$ 时菜肴数比顾客数还少，显然是不满足要求的；当 $|E| = |V|$ 时我们可以给无向边定向构造出一棵基环外向树（每个点的入度均为 1），然后每道菜供应给其顶点入边所对应的顾客；而当 $|E| = |V| - 1$ 时我们定根构造出一棵有根树，然后每位顾客选择其对应边的两个顶点中父亲节点的菜肴即可。

现在我们的任务就是找到最小边集 E 满足 $|E| > |V|$，则答案为 $|E| - 1$。可以发现最小边集 E 必然满足 $|E| = |V| + 1$，否则我们可以直接去掉一条边。而且点集中不能存在度数为 1 的点，否则我们直接去掉这个点和与其相连的边即可。由于 $\sum_{x \in V} \deg(x) = 2|E| = 2|V| + 2$，而又有 $\forall x \in V, \deg(x) \ge 2$，所以点集 V 只有两种可能：两个顶点度数为 3，其余顶点度数为 2；一个顶点度数为 4，其余顶点度数为 2。那么这个连通图的结构只有以下三种：

- 两个度数为 3 的点之间有三条不相交的路径；
- 两个不相交的环之间有一条简单路径连接；
- 两个环在一个顶点处相交。



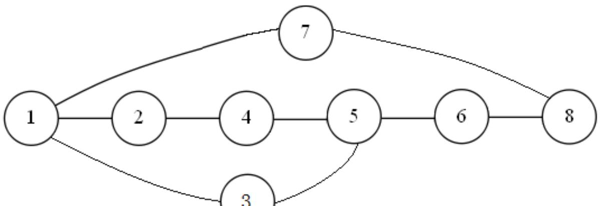


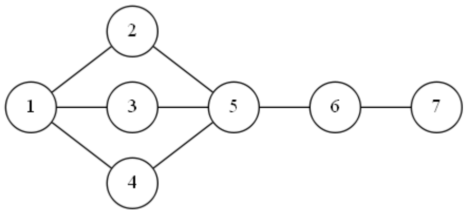


对于结构 1，我们可以枚举路径的起点 S 和终点 T，然后用 BFS 找到 S 到 T 的三条简单路径。根据 BFS 的性质，此时我们找到的必定是最短的三条路径，那么我们把三条路径的长度加起来更新答案即可。当然，三条路径有可能相交而不符合上面的结构，但其实并没有关系，我们必定可以在这个子图的基础上,通过改变起点和终点来得到三条不相交路径的。

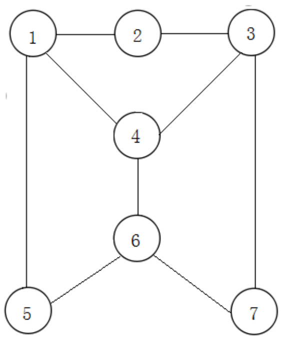
下左图起点 S＝1，终点 T＝8，三条路径 1-7-8，1-2-4-5-6-8，1-3-5-6-8 是有公共部分的。但我们认为 S＝1 而 T＝5 的话，路径 1-7-8-6-5，1-2-4-5，1-3-5 就不会相交了。

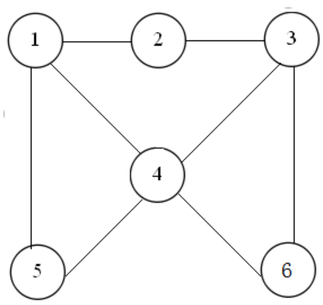
下右图起点 S＝1，终点 T＝7，我们直接去掉 5-6 和 6-7 两条边，令 S＝1,T＝5 即可。





对于结构 2，我们枚举点 x 作为连接两个环的简单路径上的某一个点，然后我们从点 x 开始 BFS 来建出原图的一棵生成树。设 d[i]表示点 i 在生成树中的深度，那么我们就是要找到两个环 P 和 Q（设 u,v 分别是环 P 和环 Q 中深度最浅的点），使得 length(P) + d[u] + length(Q) + d[v]最小。很容易证明，这两个环都必定只有一条边不在生成树中。所以我们枚举所有不在生成树上的边，求出对应环的长度和对应点的深度，更新最大值和次大值即可。当然，这两个环也可能相交，但是如果两个环有公共边，我们必定就可以找到这个边集的一个子集使其满足结构 1，如果只有公共点也可以化成结构 3。如下图：





而结构 3 基本与结构 2 相同，只要枚举两个环的交点即可。

4 复杂度分析

结构 1：枚举起点与终点 $O(n^2)$，BFS 求路径 $O(n+m)$，复杂度 $O(n^3+n^2m)$；

结构 2 与结构 3：枚举中间点 $O(n)$，BFS 建树 $O(n+m)$，枚举每条边求环的长度 $O(mn)$，复杂度 $O(n^2m)$。

所以，总复杂度为 $O(n^3+n^2m)$。

CKROACH 解题报告

1 题目大意

有 n 只虫子和 m 种杀虫剂，第 j 种杀虫剂的价格为 C_j，使用了第 j 种杀虫剂后，第 i 只虫子有 $P_{i,j}\%$ 的概率死亡。要求选择一种购买杀虫剂的方案，使得每只虫子的死亡概率至少为 90%，并且花费尽可能少。

数据范围：$50 \le n, m \le 200$

2 评分方式

对于每组数据，记 $rate = \dfrac{\max(basic - your, 0)}{basic}$，其中 $your$ 是你给出集合的总花费，$basic$ 是一个简单贪心算法给出集合的总花费，该数据你的得分是 $\dfrac{e^{rate}-1}{0.95}$，测试点的得分为该测试点每组数据的得分之和。

3 题目分析

令 $w[i][j] = -\log_{10}(1-\dfrac{p[i][j]}{100}) = 2 - \log_{10}(100 - p[i][j])$，则第 i 只虫子死亡概率至少为 90%，即 $\prod_{j \in S}(1-\dfrac{p[i][j]}{100}) \le \dfrac{1}{10}$，等价于 $\sum_{j \in S} w[i][j] \ge 1$，所以问题转化为：

最小化 $\sum_{i=1}^{m} c_i x_i$

约束条件 $\forall i \in [1,n], \sum_{j=1}^{m}(w[i][j] \cdot x_i) \ge 1$

变量取值 $x_i \in \{0,1\}$

这是一个 0 / 1 背包问题，而背包问题是 NP - hard 的，所以很难得到问题的最优解，我们考虑用贪心、随机化等算法来求一个近似解。

4 贪心算法

定义 $f[i]=\sum_{j=1}^{n} w[j][i](\text{当前虫子}j\text{的死亡概率不足}90\%)$ 来表示当前第 i 种杀虫剂的贡献值，然后我们采取这样的贪心过程，每次根据某种策略选择一种当前还有贡献的杀虫剂，直到所有虫子的死亡概率均不小于 90%。题目给出的 *basic* 贪心程序选择的策略即为从 1 到 m 依次选择，下面给出三种贪心策略：

- 每次选择花费最少的杀虫剂
- 每次选择贡献值最高的杀虫剂
- 每次选择性价比(贡献值 / 花费)最高的杀虫剂

很明显第三种贪心策略更加优秀，实际上直接使用该策略便可以在 Codechef 上获得 0.781 分，而在 Tsinsen 上获得 94 分。

5 随机化算法

最简单的算法当然就是直接随机每种杀虫剂是否使用，然后判断合法性来更新答案。但效果并不是很好，在 Codechef 上得分仅为 0.093 分。

我们还可以随机一个顺序，然后使用上面的贪心过程，并多次随机取最优解。经测试，随机 50 次可以在 Codechef 上获得 0.351 分。

当然，采用更高级的随机化策略，比如模拟退火算法，经过选择合适的邻域与参数后效果是很好的。我的找邻域方法是根据温度确定一个数 K，然后随机选择 K 种杀虫剂将它们是否购买的状态反转。结合上述贪心算法后，可以在 Codechef 上获得 0.86 分，且在 Tsinsen 上获得 108 分。