最优决策 解题报告

宁波市镇海中学 邹逍遥

1 试题大意

两个人玩这样一种游戏：

1.双方各有一个能量槽，可以存储0~n的能量值，当能量槽满了的时候可以释放大招，消耗所有能量并秒杀一切技能（假如两个人都放大招则游戏继续）。初始时双方能量都是0。

2.除了大招之外有m个小技能，小技能有3类：

- (1):消耗型，消耗x个能量，只有能量不小于x时才能使用。
- (2):免费型，不消耗能量，任何时候都可以使用。
- (3):补给型，使用后能获得x个能量，但是获得后最多只能有n个能量（多出来部分的会浪费掉），任何时候都可以使用。

3.小技能之间有相克的关系，当然也有不相互影响的小技能对。

4.每回合每个人必须选择一种可行技能，然后同时亮出。假如这两个技能相克，则某一方立即获得胜利。否则双方更新自己的能量然后继续游戏。

两个人需要分别制定一张决策表，即事先写明在某个状态（己方x能量对方y能量）自己出每个决策的概率，然后根据决策表进行游戏，游戏过程中不能修改决策表。

你需要解决三个子问题：

- Subtask1:给出游戏局面和双方的决策表求出胜率。
- Subtask2:给出游戏局面和某方的决策表，找出一个决策表使得它对战这个决策表胜率尽量高。

- Subtask3:给出游戏局面，找出一个决策表使得它对战任意决策表胜率都不低于50%。

数据为随机生成。
数据保证满足以下性质：

- 三种技能都至少有一个
- 每个补给型技能至少被一个技能克制
- 至少存在一个补给型技能使得只有消耗型技能才能克制它
- 数据中给出的决策表中每一项都不为0。

2 数据规模

No表示测试点编号，Case表示数据组数，type表示subtask编号。

No.	Case=	n=	m=	type=	备注
1	1	5	30	1	输入就是样例四
2	100	5	30	1	
3	100	3	5	2	
4	30	5	30	2	
5	1	3	3	3	游戏局面与样例一相同
6	1	5	3	3	游戏局面与样例一相同
7	100	2	5	3	
8	100	5	5	3	
9	100	2	30	3	
10	10	5	30	3	

3 算法介绍

3.1 Subtask 1

3.1.1 针对测试点一的做法

由于给出了第一个点的输入数据，可以写一个模拟程序进行多次模拟后即可得出答案。

3.1.2 满分做法

将己方每个状态的胜率设为n^2个变量，那么可以列出n^2个方程，利用高斯消元解出这些方程即可得到答案。

设己方i能量，对方j能量的变量为$F(i, j)$，己方在(i, j)状态时选择k决策的概率是$P(i, j, k)$，对方是$Q(i, j, k)$，双方分别使用决策k和决策l时将转移到$T(i, j, k, l)$（假如相互克制则转移到必胜态或必败态），那么方程为：

- $F(i, j) = \Sigma_k \Sigma_l P(i, j, k) * Q(j, i, l) * F(T(i, j, k, l))$
- $F(\text{必胜态}) = 1$
- $F(\text{必败态}) = 0$

因为题目保证出每个可行决策的概率都不为0，所以不会进入死循环，只需要直接解出胜率即可，不需要判断任何特殊情况。

时间复杂度$O(n^6 + n^2m^2)$。

3.1.3 另一种做法

由于题面保证了数据随机生成，也就是说游戏期望进行的轮数不会很多，经过测试，在几十轮以后几乎可以认为双方胜率均为50%。

于是我们可以使用迭代进行求解。每次用下一轮的胜率来更新这一轮的胜率。

即$F(i, j) = \Sigma_k \Sigma_l P(i, j, k) * Q(j, i, l) * F'(T(i, j, k, l))$，其中$F'$表示下一轮的胜率。

由于m^2n^2和n^6差不多大，运行时间和高斯消元法没有太大区别。

3.1.4 得分估计

这个Subtask基本属于送分，预计绝大多数选手能拿到满分。

3.2 Subtask 2

3.2.1 针对测试点三的做法

定理1. *在对手每个决策都有大于0的概率选择的情况下，存在一种最优解使得每个状态一定是100%使用某个技能。*

证明：假如有一种最优的决策表使得某个状态并不是100%使用某个技能，那么首先可以计算出每个状态的胜率，接着可以算出那个状态最优的决策。

考虑将那个状态的决策替换为全部使用最优决策，假如最优决策和原来有概率选的所有决策胜率都相同那么胜率没有任何变化，所以是可行的。

否则考虑进行多次迭代。首先那个状态的胜率会上升，那么迭代后会影响到的状态就是每个会转移到它的状态。容易看出那些状态的胜率也都会上升。所以没有一种状态的胜率会下降。

那么只有一种情况：表面胜率上升但是真实胜率下降了。这样只可能在改变之后进入了死循环，因为在死循环中无论这些状态的胜率是多少都不会推导出错误，但是由于死循环时会算作平局，所以实际上它们是0.5。于是在这种情况下会使胜率下降。但是由于对手每个决策都有大于0的概率选择，所以这种情况不会出现。这样就保证了在题目条件下一定存在这样一种方案。

那么可以枚举每个状态使用的技能然后用上一问的做法进行检测。时间复杂度$O(m^{n^2} * (n^6 + n^2m^2))$。

3.2.2 满分做法

由于题面保证了数据随机生成，也就是说游戏期望进行的轮数不会很多。

于是我们仍然可以使用迭代进行求解。每次根据下一轮的胜率找出这一轮的最优决策并算出这一轮的胜率。

设状态(i, j)的胜率为$W(i, j)$上轮迭代中状态(i, j)的胜率为$W(i, j)$那么$W(i, j) = \max_k(\Sigma_l Q(j, i, l) * W'(T(i, j, k, l)))$。

经过测试大约迭代50轮左右就可以满足精度要求，时间复杂度为$O(n^2m^2\times$迭代轮数)。

3.2.3 得分估计

这个Subtask也比较简单，只需要观察到定理1这样的性质就可以解决。预计能有一半左右的选手拿到满分。

3.3 Subtask 3

3.3.1 针对m比较小的数据

使用调整法。

类似Subtask 2，每次把某个状态的当前最优决策的概率提高，减小其他决策的概率，并更新胜率。适当调整参数可以拿到更多的分。

在m比较小的时候比较容易拿分，但是m比较大的时候由于精度要求和多组数据比较难拿到分。

3.3.2 针对n比较小的数据

首先考虑只有一种状态时的情况。此时题目变成了：一个对称的矩阵，第一个人给出选每一行的概率，第二个人针对性地挑选某一列，最后第一个人的收益就是对应的格子里的数，第二个人的收益就是那个数的相反数。当矩阵里的数都是0,-1,1 时，此题就和BJOI2011Day2《读心》相同。

有两个数据$n = 2$，由于题面保证了至少有一个补给型技能使得只有消耗型技能才能克制它，所以当1:0时必胜，0:1时必败，0:0和1:1平局。

已知了每个状态的胜率后每个状态就可以独立开来分别求解。

我们可以将每个转移到必胜态的收益指定为1，转移到必败态的收益指定为-1，转移到平局态的收益指定为0，那么就可以利用《读心》的解法来解决这两个点。

《读心》的官方解法使用的是调整法，这里介绍另外一种更加靠谱的线性规划解法。

3.3.3 《读心》的线性规划解法

首先回忆一下什么是线性规划问题：

求出一组**非负**实数$x_1 \sim x_n$，满足约束$\Sigma a_{0,i} * x_i \le r_0, \Sigma a_{1,i} * x_i \le r_1 \cdots$，最大化$\Sigma z_i * x_i$。

使用单纯形法可以比较方便并且快速地解决线性规划问题。关于单纯形法的介绍详见《算法导论》。

由于当前状态的胜率已知（为0.5），所以可以比较容易地列出线性规划方程：

- 线性规划中有m个变量，分别表示每个决策的出现概率。最大化m个变量之和。
- 这样可以列出m个不等式，分别表示对手选择第i个决策后对方的收益不大于0。
- 可以证明最后m个变量之和一定为1，那么解出该线性规划后最后每个变量的值就是方案。

利用这个解法就可以解决n比较小的数据。

3.3.4 将线性规划用于本题

由于题面保证了数据随机生成，也就是说游戏期望进行的轮数不会很多。

那么可以使用迭代，即用假设已知的每个状态的胜率去更新新的胜率。

我们可以二分新的胜率，然后检测它是否符合条件。线性规划的列法基本相同：

- 线性规划中有m个变量，分别表示每个决策的出现概率。最大化m个变量之和。
- 这样可以列出m个不等式，分别表示对手选择第i个决策后自己的胜率不高于二分中算出的胜率。（这里计算胜率时使用$1-\Sigma$某种决策对出现概率$\times$该决策对转移到的状态的对手的胜率，所以只减小某个变量后得到的解依然满足这些约束，即假如最后所有变量之和大于1也一定存在可行方案）
- 假如最后m个变量之和大于等于1，那么就表示这个胜率是可达到的，并且方案就是最后每个变量的值。

经过测试大约50轮左右就可以满足精度要求。

我们可以认为最后一轮迭代后的胜率就是正确的胜率，于是最后再对每个状态跑一遍线性规划就能求出决策。

3.3.5 满分做法

类似上一个做法，只要把线性规划换成如下形式就能一次解出胜率从而省掉二分的log。

- 线性规划中有$m+1$个变量，分别表示每个决策的出现概率和该状态的胜率。最大化胜率。
- 首先可以列出两个不等式使前m个变量之和等于1。
- 然后可以列出m个不等式，分别表示对手选择第i个决策后自己的胜率不低于 变量$m+1$ 。

时间复杂度为$O(n^2 \times$解$m*m$的线性规划时间$\times$迭代轮数)。
这样就可以通过全部数据。

3.3.6 得分估计

预计能有0~2名选手拿到满分，4~6名选手拿到20分及以上。