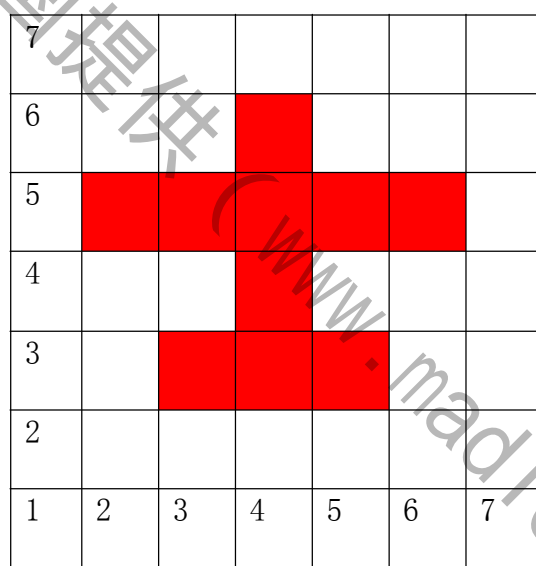


飞机对战游戏

一、问题重述

在游戏前双方各准备一张坐标纸，在上面分别制作 7×7 的方格，如图 1 所示。在自己的方格中画一架飞机，飞机呈“士”字形，其中上面的一长横占 5 个格子，下面的短横占 3 个格子，一竖占 4 个格子，最上面突出的一个格子代表机头。所画飞机的位置以及机头的指向由游戏者自己决定，游戏结束前双方不能互看对有一种在学生中间比较流行的双方对战游戏方的坐标纸。游戏时双方交替用“炮弹”打击对方，攻击的一方报告“炮弹”打击的位置，被攻击的一方报告是否命中飞机。例如：被攻击方的飞机画法如图 1 所示，攻击者报告“炮弹”的打击位置是 $(4, 3)$ ，从图中可知，“炮弹”恰好落在飞机所在的红色格子上面，被攻击方报告飞机被击中，接下来刚才的被攻击方变成攻击方进行上面的攻击步骤，双方交替攻击对方，如果某一方被命中机头，游戏结束，被命中机头的一方失败。游戏双方都在通过打击后对方的反馈信息来猜测对方飞机的位置。

游戏比赛采用 19 局 10 胜制。



问题一：设计一个人机对战的“飞机对战”游戏。要求先由计算机进行攻击，以取胜为目标，给出进行游戏的策略。

问题二：考虑在 9×9 坐标纸上画两架飞机的游戏方式，两架飞机所占的格子不能重合，游戏方法同上。其中一架飞机被命中机头时要报告有一架飞机被击落。当某方的两架飞机都被击落时游戏结束，被击落方失败。分析这种游戏方式与只画一架飞机的游戏方式在策略上的不同点。

问题三：如果将问题二中的游戏方式设计为一个网络游戏，由三个真人对战，三人轮流作为攻击方，攻击方可以选择另外两方之一作为攻击对象，每次只能发射一发“炮弹”。如果某一方的两架飞机均被击中，他将退出这一局游戏，另外两方仍将继续游戏直到二者决出胜负。打中机头可以得 1 分，打中机头并把被攻击方踢出局可以得 3 分，打中飞机其它部位或者未击中不得分，比赛采用 18 局，最后总得分（累加每局得到的分数）最高的一方获胜，如果出现平分，加赛一局决定胜负。考虑每局都首先由你作为攻击方，设计一套游戏策略，使你能在比赛

中取胜。

注：假定游戏中，不存在任意两方联合的可能。

二、问题简析

§ 问题一：

为了使计算机获胜，则应该使打击获得的效益最大。本题的关键是建立决策树。

如果计算机的策略是在某步中选择打击 7×7 个方格中可能出现机头的概率最大的点，则在该步下获胜的概率最大，下一步得到 3 种反馈——获胜、仅击中机身以及未击中机身。获胜则游戏结束，若未获胜则根据将所有的飞机分为 2 类：

A：打击的点在飞机上，且打击点不是机头；

B：打击的点不在飞机上。

若仅击中机身，则在 A 类飞机中寻找机头出现概率最大的点，作为下次的击打目标。同样的，若未击中机身，则在 B 类飞机中寻找机头出现概率最大的点，作为下次的击打目标。如第一步选择打击点 $(4, 4)$ （以 $(4, 4)$ 点作为机头的飞机一共有 4 种，其他点均小于 4 种），若反馈获胜，则可知点 $(4, 4)$ 为人的飞机机头；若反馈，仅击中机身，则可知 $(4, 4)$ 点在飞机上，且 $(4, 4)$ 不是飞机的机头，属于情况 A。第二步就要在 A 集合中寻找机头出现概率最大的点作为打击目标；若反馈未击中机身，则 $(4, 4)$ 不在飞机上，属于情况 B。第二步就应该在集合 B 中寻找机头出现概率最大的点继续打击。

但是经过研究发现，该种策略下某步中飞机机头出现概率最大的点经常不唯一。遇到这种情况，则可以随机的选择一个概率最大的点做为打击目标，（注：实验中先假设可以随即选择，并且所选择不影响最终的结果，经过了计算机模拟和进一步分析，发现，随机的选择一个点做为打击目标，只对下一步要打击的点有影响，而对整个的系统无影响）。

§ 问题二：

题设改为在 9×9 坐标纸上画两架飞机进行游戏，并且两架飞机所占的格子不能重合。由于存在飞机的数量增大，对模型的求解造成更多的影响。

游戏结束的可能情况不同，可以分为如下几种：一人两架飞机被击落，另一人都未被击落的情况；一人先被击落一架飞机，后又对方两架连续击落；一人先被击落一架飞机，后将对方一架击落，最后又未能获胜。

问题二可以是对问题一的扩展，但信息量增大。所需要的反馈信息也增加，除了问题 1 中的的是否机中飞机的机身或机头，还需要包括 9×9 坐标纸上所存在的飞机的个数

§ 问题三：

当游戏设计成三人对战时，己方获胜的条件是要求在 18 局总得分要比另外两个参与者高。则易得，每局开局时己方选择打击的对手是当前与自己分数较为接近或是更高的参赛者，而在比赛中每步选择打击的对手，一般情况下是让己方获得更高的分数期望的参赛者。每步选择打击的目标点仍然参照问题二建立的模型，进行信息的筛选。

三、模型假设

经过的简要分析后，得到基本假设：

假设一：计算机是以获胜为目的，并尽最大可能获得胜利；

假设二：人的选择是随机的，既是个随机变量；

假设三：计算机先走；


假设四：在建立决策树的时候，是不考虑人取胜结束游戏，以计算机的胜利为结束；


四、模型建立及详细分析

※问题一：

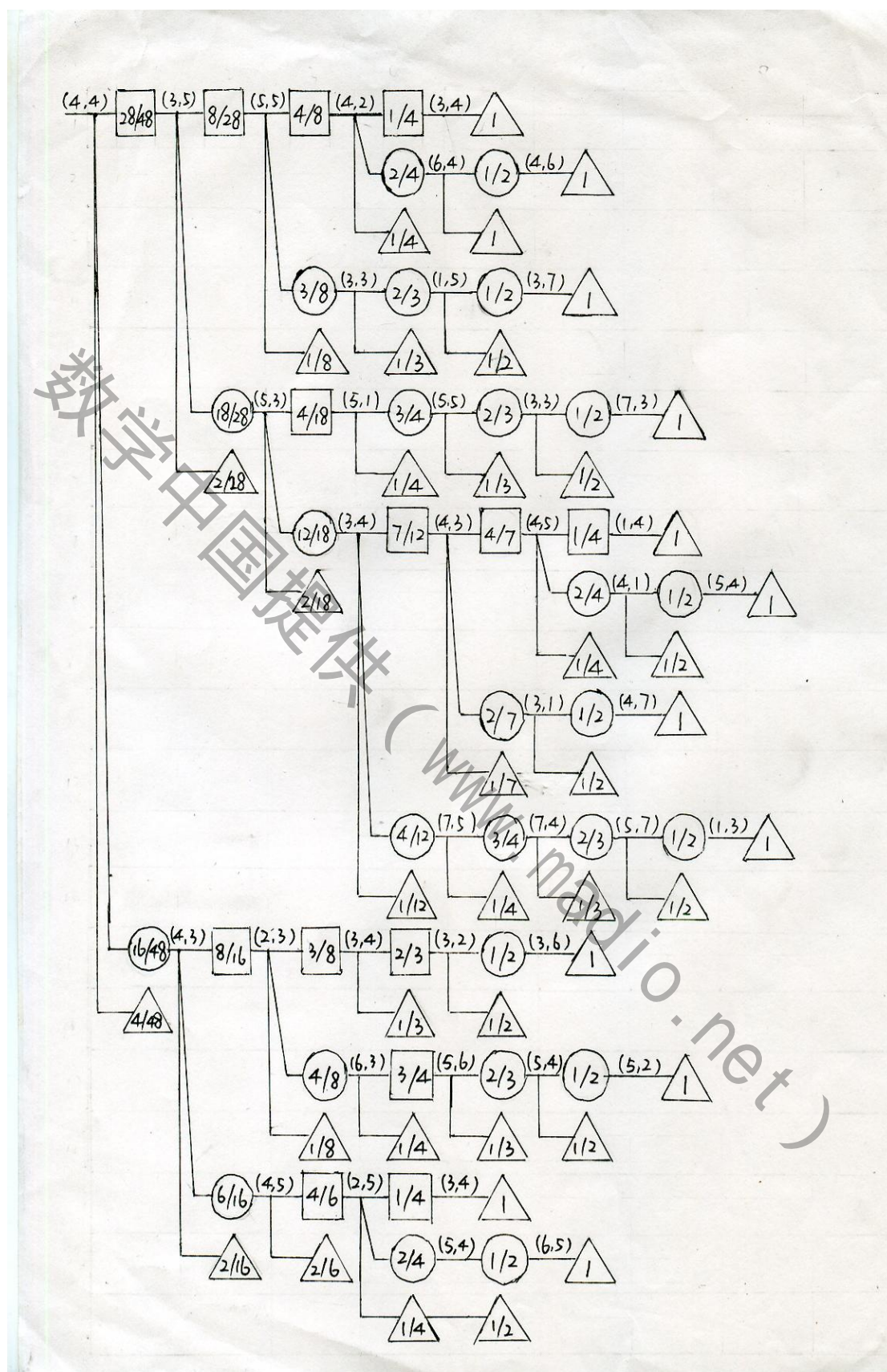
根据假设和进一步分析，通过计算机编程把所有的情况考虑进去后，我们在遇到可以选择多个等概率打击目标的情况下，随机的选择一个，由计算机编程实现（程序见附录）。

画出的决策树如下：

图形说明：所选择的打击目标的不在机身的路径，用圆形表示。图中为 

所选择的打击目标的在机身且不为机头的路径，用方块表示，图中为 

所选择的打击目标的为机头，既获胜的路径用三角表示，图中为 



上图中的决策树的建立时应参照一下条件:

所有图形中的分数表示该路径的概率, 特征如下:

A) 在一个分支上所有分支的和为 1

B) 某层次上的分支的路径概率的分母为上一层路径概率的分子

路径上的点表示在该步下选择打击的目标。

例如：以 (4, 4) 开始的三条分支的和为 1。如果打中的是机身且不是机头则进入分支 28/48，而由该点出发的各分支概率又以 28 为分母。并且各点遵循如上点的规则，继续由打击的信息进行判断。

将决策树模型按照获胜步数与各步数下获胜的情况总数的关系建立表格（一），如下：

步数 n	1	2	3	4	5	6	7	8
M(n)	4	4	6	7	10	9	6	2

表格（一）

$$E = \sum_{n=1}^8 n * m_n = (1*4 + 2*4 + 3*6 + 4*7 + 5*10 + 6*9 + 7*6 + 8*2) / 48 = 4.58$$

以上是建立了一个决策树，现在验证他的可行性，即计算出计算机获胜的概率。将模型进一步完善。

模型 I (i):

假设一：人知道飞机的头可能出现的位置是 33 个点；

假设二：人的选择是随机的，在那 33 个点中随机选择；

假设三：人是有记忆的，选择过的点可能再继续选；

假设四：取消基本假设五，即考虑人获得胜利对计算机获得胜利概率计算的影响；

假设五：计算机先走；

假设六：将人选择某点作为机头的概率考虑进去；

由于人选择打击的点是随机的，故计算机机头位置对计算机获胜的概率没有影响，所以可以不考虑计算机的机头的位置。

符号说明：

An：计算机在第 n 步获胜的概率

Bn：人在第 n 步获胜的概率

Mn：为表格（一）中的数据，即 n 步获胜的情况的个数

P0：计算机第一步获得胜利的概率。由决策图中不难发现，P0 只和相应步长以及飞机总数有关，即 $P0 = M_n / 48$

P1：该步计算机不获胜的情况下。人获胜的概率，由于，计算机的飞机头是在这个 33 个格子中，而人选择这 33 个格子，又因为计算机放置机头的位置和人选择的位置是相互独立的，所以一共可能情况为 $33*33$ ，而飞机的放置和人的选择可以确定打中的机头为 48 种，既打中机头的概率 $P1 = 48 / (33*33) = 0.044$

模型分析：

人是有记忆的，等同于概率中放回抽样模型，即前一次人的选择对其后一次没有影响，则人每次的选择打击的点的概率相同。

计算机在第一步获胜的概率 $A1 = 4/48$

人在第 n 步获胜的概率

$$B_n = \left(1 - \sum_{i=1}^n A_i - \sum_{i=1}^{n-1} B_i \right) \times p_1 \quad (\text{i-1})$$

计算机在第 $n+1$ 步获胜的概率

$$A_{n+1} = \left(1 - \sum_{i=1}^n A_i - \sum_{i=1}^n B_i \right) \times p_0 \quad (\text{i-2})$$

将方程组联立化简可得：

$$\frac{B_n}{P_1} - \frac{A_{n+1}}{P_0} = B_n \quad (\text{i-3})$$

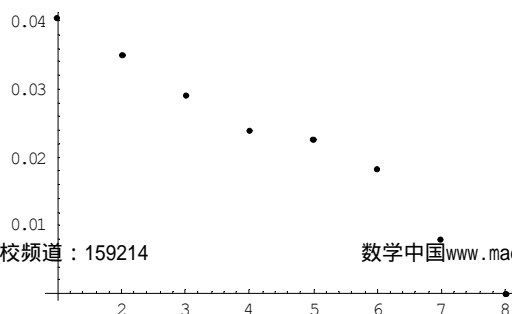
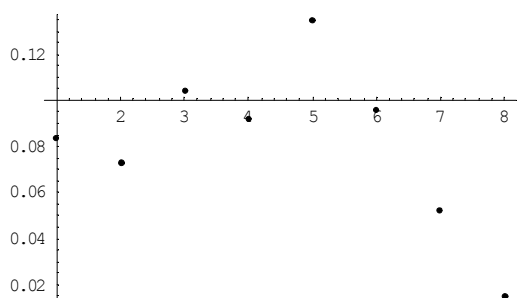
$$A_{n+1} - A_n = -P_0 A_n - P_0 B_n \quad (\text{i-4})$$

将等式 (i-3) (i-4) 经过 MATHMATIC 模拟后，得到概率的数据如表格 (二)：

步数	A_i	B_i
1	0.0830	0.0437
2	0.0730	0.0350
3	0.1040	0.0290
4	0.0920	0.0238
5	0.1345	0.0225
6	0.0959	0.0183
7	0.0519	0.008
8	0.0150	0
总数	0.6483	0.1803

表格 (二)

并且由所列数据画出的散点图可清楚看得人机双方的获胜概率改变趋势：



计算机获胜的概率随步数 n 的变化

人获胜的概率随步数 n 的变化

模型 I (ii):

假设一：人在 7×7 的方格图中随机选择攻击位置；

假设二：人是很有经验的玩家，即人采取的打击策略和计算机一样；

假设三：取消原来的假设五，且考虑人可以在计算机之前就获胜；

模型建立与 (i) 相同，因为假设也知道怎样获胜，则计算机打击何点，人也打击何点，此时的 p_1 既为上次计算机获胜的概率，既第 N 次人获胜的概率和第 N 次计算机获胜的概率相同， $A_1=4/48$ ；

人在第 n 步获胜的概率：

$$B_n = \left(1 - \sum_{i=1}^n A_i - \sum_{i=1}^{n-1} B_i \right) \times p_1 \quad (\text{ii-1})$$

计算机在第 $n+1$ 步获胜的概率：

$$A_{n+1} = \left(1 - \sum_{i=1}^n A_i - \sum_{i=1}^n B_i \right) \times p_0 \quad (\text{ii-2})$$

$$p_0 = m \div 48$$

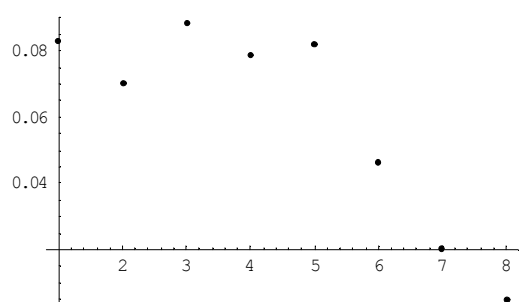
$$p_1 = p_0$$

在 MATHEMATICA 环境中，用等式 (ii-1)、(ii-2) 进行迭代求解（代码见附录），可以得到人和计算机在每一步获胜的相应概率，所得数据如表格（三）：

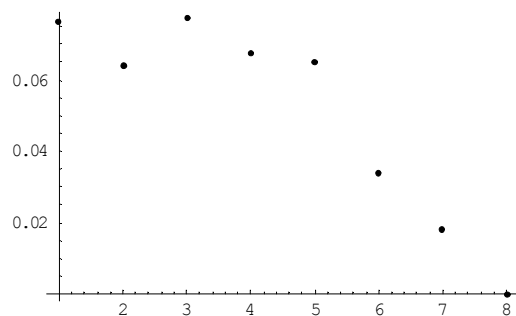
步数	A_i	B_i
1	0.0830	0.0764
2	0.0701	0.0642
3	0.0883	0.0773
4	0.0788	0.0674
5	0.0820	0.0651
6	0.0464	0.0377
7	0.0204	0.0179
8	0.0050	0
总数	0.4740	0.4060

表格（三）

并且由所列数据画出的散点图可清楚看得人机双方的获胜概率改变趋势：



计算机获胜的概率随步数 n 的变化



人获胜的概率随步数 n 的变化

由表格（二）、（三）比较可得：

人随机选择点打击，即只在可能存在飞机的 33 个点中选择的情况下计算机获胜的概率会比人知道怎么样可以获胜的时候计算机获胜的概率大。可以概括为：人的智商越高，则计算机在赢的概率越低，但是计算机赢的概率始终比人获胜的概率大。

下面计算人机对战 19 场，计算机获得 10 场胜利的概率。因为 19 比较大，则可看作基本符合二项分布规律。令进行总盘数为 N ， $P\{N=n\}$ 为进行 n 场对战后获胜的概率：

$$P\{N=n\} = C_{n-1}^9 \times p^{10} \times q^{n-10}$$

其中 p 、 q 分别为计算机和人一局获胜的概率，所以计算机 19 局 10 胜的概率：

$$P = \sum_{i=1}^n P\{N=i\}$$

对于模型 II(i)，有下表：

进行 n 场	10	11	12	13	14	15	16	17	18	19
$P\{N=n\}$	0.08560	0.1863	0.2229	0.1940	0.1372	0.0836	0.0455	0.0226	0.01046	0.00455

表格（四）

由上表数据代得：

计算机获胜的概率 $P=0.99275$

对于模型 II(ii)，有下表：

进行 n 场	10	11	12	13	14	15	16	17	18	19
$P\{N=n\}$	0.0021	0.0095	0.0241	0.04439	0.06656	0.0860	0.0992	0.1046	0.10257	0.0946

表格（五）

由上表数据代得：

计算机获胜的概率 $P=0.6336$

综上所述：即使人的智商再高，计算机获胜的概率都要比人获胜的概率大。
所以计算机打击目标的策略可行。

※问题二：

问题二将 $7*7$ 的格子改成了 $9*9$ 的，并且又加入了一架飞机，使得飞机总的可能排列数变大。两架飞机带来的变化在于问题一中只要你打中了飞机就已经获胜，而问题二中打中一架飞机还得继续打中第二架才能获胜。故在问题二中新引入反馈参数 P_n ，表示图中剩余的飞机数。在策略上仍然采用问题一中的策略，即打击飞机头出现的最大概率的点。

问题一中，每打击一次，得到的反馈为打中机头获胜、打中飞机机身或者未打中飞机，而问题 2 中，选择打最大概率点后，得到的反馈为打中机头获胜、打中飞机机身、未打中飞机以及图中剩余飞机数目。若反馈得到图中仅剩一架飞机，则问题二又回归到问题一上直至游戏结束。

模型 II：

假设一：计算机有足够的计算能力能处理庞大的反馈信息；

假设二：人的选择是随机的；

假设三：在计算选择击打路径的时候不考虑人会在计算机在人获胜之前获胜；

假设四：人没有强大的计算能力，人是不知道哪个点概率大，即是在机头出现的位置中随机选则；

由于人选择打击的点是随机的，故计算机机头位置对计算机获胜的概率没影响，所以可以不考虑计算机的机头的位置。

符号说明：

i ：所选择打的点的横坐标；

j ：所选择打的点的纵坐标；

P_n ：击打每一点时所存在的飞机总数；

$F(\text{head}, \text{body}, \text{none})$ ：选择击打 (i, j) 点得到的反馈信息。 $\text{head}=0$ 表示没有击中机头； $\text{head}=1$ 表示此时打中一个机头； $\text{head}=2$ 表示第二次打中机头。 $\text{body}=0$ 表示没有击中飞机机身； $\text{body}=1$ 表示击中飞机机身。 $\text{none}=0$ 表示此时击中了飞机机头或者机身， $\text{none}=1$ 表示没有击中飞机。例如： $(0, 1, 0)$ 表示打中的是机身没打中机头， $(0, 0, 1)$ 表示没有打中飞机的任何部位。

M_f ：表示得到反馈信息后符合反馈信息的飞机可能总数。初值为 2352。

模型分析：

根据假设进一步分析，通过计算机编程把所有情况遍历了以后可得（程序见附录），比赛开始时，人所放的飞机所有可能数为 2352，人所放的飞机的两个机头（此时是等价的）所在位置数目分布和飞机所经过的点的数目分布如表格（六）和表格（七）：

0	0	65	51	44	51	65	0	0
0	0	47	30	22	30	47	0	0
65	47	72	120	106	120	72	47	65

51	30	120	148	130	148	120	30	51
44	22	106	130	112	130	106	22	44
51	30	120	148	130	148	120	30	51
65	47	72	120	106	120	72	47	65
0	0	47	30	22	30	47	0	0
0	0	65	51	44	51	65	0	0

表格（六）

表格（六）中每格中数值的意义：以该点作为飞机头的飞机的总数
其中各个格子中数值的和为 2352

0	131	299	308	280	308	299	131	0
131	332	594	647	654	647	594	332	131
299	594	956	1052	1054	1052	956	594	299
308	647	1052	1162	1056	1162	1052	647	308
280	654	1054	1056	960	1056	1054	654	280
308	647	1052	1162	1056	1126	1052	647	308
299	594	956	1052	1054	1052	956	594	299
131	332	594	647	654	647	594	332	131
0	131	299	308	280	308	299	131	0

表格（七）

表格（七）中每个点的意义为：机身（包括机头）经过该点的飞机的个数

计算机先打击机头概率最大的点，遇到有概率一样的点的情况，则随机选择其中一个，类似于问题一的策略。下面先解释第一个点的选择，以方便解释模型。第一次击打选择飞机机头可能在的概率最大点（4，6），且此时 P_n 的值为 2，得到反馈向量 F (head, body, none) 可能为（1，0，0）、（0，1，0）或（0，0，1）分别表示击中一架飞机的机头、击中飞机的机身以及未击中飞机。

$F=(1, 0, 0)$ 时，修改 $P_n=1$ ， $M_f=148$ ，此后类似于问题一模型。下一步如问题一的打击策略，在 $M_f=148$ 的情况下统计可能以该点作为飞机头的飞机个数，找出最大的点，作为下次的打击目标。计算机计算结果如下：

0	0	0	0	0	2	3	0	0
0	0	0	0	0	1	1	0	0
0	0	0	2	1	2	3	2	2
0	0	2	0	3	7	4	1	1
0	0	1	3	2	4	6	2	2
2	1	2	7	4	8	6	2	3
3	1	3	4	6	6	6	2	4
0	0	2	1	2	2	2	0	0
0	0	2	1	2	3	4	0	0

可知下一步将选择打击点 (6, 4)。

$F(0, 1, 0)$ 时, $P_n=2$, 修改 $M_f=1014$, 此时 P_n 仍等于 2, 即还没有打下任意一架飞机, 仍和问题二模型一致。下一步在 $M_f=1014$ 情况下统计可能以该点作为飞机头的飞机个数, 找出最大的点, 作为下次的打击目标。计算机计算结果如下:

0	0	65	51	44	7	15	0	0
0	0	3	30	1	7	15	0	0
65	3	72	26	67	40	46	8	15
51	30	26	0	49	50	49	7	14
44	1	67	49	78	68	37	6	14
7	7	40	50	68	56	48	14	20
15	15	46	49	37	48	26	22	24
0	0	8	7	6	14	22	0	0
0	0	15	14	14	20	24	0	0

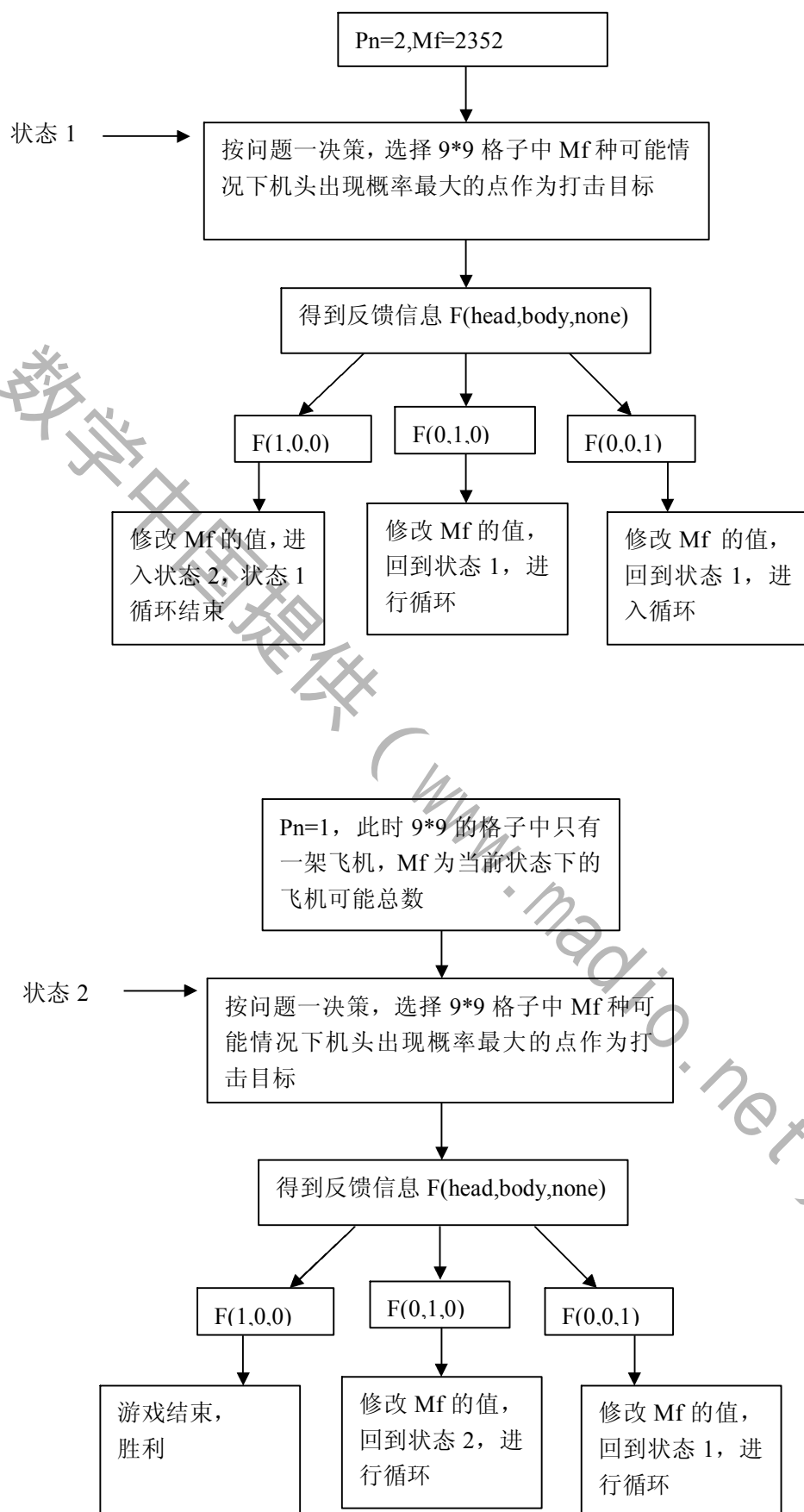
可知下一步将选择打击点 (5, 5)。

$F(0, 0, 1)$ 时, $P_n=2$, M_f 修改为 1190, 此时 P_n 仍等于 2, 即还没有打下任意一架飞机, 仍和问题二模型一致。统计可能以该点作为飞机头的飞机个数, 找出最大的点, 作为下次的打击目标。计算机计算结果如下:

0	0	0	0	0	42	47	0	0
0	0	44	0	21	22	31	0	0
0	44	0	92	38	78	23	37	48
0	0	92	0	78	91	67	22	36
0	21	38	78	32	58	63	14	28
42	22	78	91	58	84	66	14	28
47	31	23	67	63	66	40	23	37
0	0	37	22	14	14	23	0	0
0	0	48	36	28	28	37	0	0

可知下一步将选择打击点 (3, 6)。

通过以上的举例可以清晰的得出模型的建立过程, 建立的流程图如下:

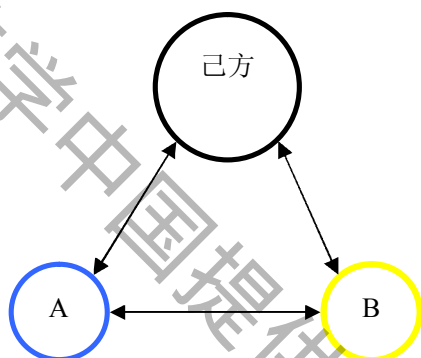


该模型选择的是每次打当前飞机最大概率点，先判断 P_n ，当 $P_n=2$ 时，即 2 架飞机都还没摧毁时，根据反馈向量 $F(\text{head}, \text{body}, \text{none})$ 可以判断、删选出所有符合反馈向量的飞机。

问题一、二建立的决策**根据反馈信息动态选择**最大概率点，通过打击机头概率最大点来获得最大效益。

※ 问题三：

由题目可得：



图形说明：

三个游戏者两两都存在攻击与被攻击的关系， \rightarrow 指向的一方为被攻击的对象；

在每次攻击时，只能选择敌方二者之一进攻，因此图中六个箭头不可能同时存在；

游戏中没有结盟的约定，因此每一轮游戏存在的箭头会变化；

当有一方的两架飞机都被击落，他将退出游戏，而此时的游戏变为问题二中的两人对战，而由于之前的战斗，二人的游戏视为在 9×9 方格中进行游戏到某步的继续。

模型 III：

假设一：游戏中的任何一方的信息都是公开的；

假设二：己方有强大的记忆力和数据统计分析能力；

假设三：在攻击过程中另两方的攻击目标选择是随机的；

假设四：游戏中不存在结盟的情况；

假设五：其他两人选择的打击点是随机的；

符号说明：

δ_i ：第 i 步由于反馈信息，对击中飞机头概率的影响系数，即反馈深度；

p_{A1} ：第一步打中 A 的机头的概率；

p_{Af} ：从第二步开始，在反馈深度的影响下的概，打中 A 的机头概率；

p_{B1} ：第一步打中 B 的机头的概率；

p_{Bf} ：从第二步开始，在反馈深度的影响下，打中 B 的机头的概率；

$Sum(A)$ ：到当前即将开始的那局游戏为止，A 方已得的分数总和；

$Sum(B)$ ：到当前即将开始的那局游戏为止，B 方已得的分数总和；

$Max(x, y)|_{A:B}$ ：选择打击对象函数。x, y 为两变量，受权值和得分总和的影响，并且当 $x > y$ 时，选择 A，反之选择 B；

模型分析：

由题设可得，打落第一架飞机能拿 1 分，即设此时权值为一，打落第二架飞机能拿 3 分，即设此时权值为 3。由于 A 和 B 双方都是随机选择目标和打击点，所以己方飞机头的位置无论在哪被对方击落的概率都相同，故自己不存在所谓的防守策略。我们把模型分为两部分，其中比赛开始的前少数局是以得分为目的，要求自己拿最高的分，并尽量抑制别人得高分。任意一个对手能使我们拿的分是一样的，因此每局比赛初始我们通过 $Max(Sum(A), Sum(B))|_{A:B}$ 即当前局分数较高的一方做为击打对象以抑制对方拿分。而在后面的比赛局中遇到 A, B 双方能获得的分权值不一致时，分为如下三种 3 种情况：

- 一、在基本平分和落后的情况下（不包括最后几局，这个因数后面考虑），选择击打的对象通过 $Max(E_a * p_{Af}, E_b * p_{Bf})|_{A:B}$ 来决定，以使己方能拿到更加高的分。我们命名这种情况为**正常型**。
- 二、在大比分领先其他良人的情况下，我们选择 $Max(p_{Af}, p_{Bf})|_{A:B}$ 来作为选择下次打击的对象的依据，命名为**保守型**。
- 三、在最后几局并且落其中某人的情况，选择我们 $Max(E_a, E_b)|_{A:B}$ 来选择下次击打的对象，命名为**搏击型**。

选择了击打的对象后，可以根据问题 2 中的模型，根据所得到的反馈情况计算出该击打的点。

例如：当 3 人的分数差别不大时或虽落后但是比赛远没结束时，若我们打 A 能拿一分的概率为 80%，打 B 能拿三分的概率为 30%，此时我们要选择的是击打 B 点。当我们大比分领先的情况下，我们选择的击打的点是 A，当最后几局比赛中我们落后，我们就要选择击打 B 就算 B 的概率为 20%也要选择 B 即为搏击只要拿高分不管概率。

■先讨论下己方攻击 A 后第一次攻击后的情况以方便建立模型；
假定己方第一次攻击的是 A，则经过一轮后的反馈信息的结果如下：

(1) A 攻击 B B 攻击 A	反馈信息包括	A: {A1, A2} (表示 A 有 2 个反馈信息) B: {B2}
(2) A 攻击己方 B 攻击己方	反馈信息包括	A: {A1}
(3) A 攻击己方 B 攻击 A	反馈信息包括	A: {A1, A2}
(4) A 攻击 B B 攻击己方	反馈信息包括	A: {A1} B: {B2}

说明：A1 为己方攻击 A 所获得的反馈信息，A2 为 B 攻击 A 获得的反馈信息。B2 为 A 攻击 B 获得的反馈信息。

若第一次攻击没有飞机被打落，即你下一步若打中飞机头带来的得分没变化：

- 处于情况（1）A 有 2 个反馈信息 A1, A2，而 B 只有一个反馈信息 B1，则 $p_{Af} = \delta_1 * \delta_2 * p_{A1}$ ，而 $p_{Bf} = \delta_1 * p_{B1}$ 。此时分析，人第一次打击 A 和打击 B 都是按照问题 2 中模型进行打击，即 2 个的概率相等，而显然 A 的反馈信息比较多，由上式可得， $p_{Af} > p_{Bf}$ 。而得分的权值没有改变。所以下次击打目标仍为 A；

- 处于情况（2），己方只有 A 的反馈信息，所以通过类似分析可得，下次击打目标仍为 A；即

- 处于情况（3），类似得，下次目标为 A，

- 处于情况（4），A 有一个反馈信息，B 有一个反馈信息，比较 $\max(p_{Af}, p_{Bf})$ ，

选择打击点，由假设五和问题 2 中的模型可得， $p_{Af} > p_{Bf}$ 。所以己方下一步击打的目标仍为 A。

假设进行了 n 步，此时第一架飞机击落，若是 A 的被击落，对打击目标没有影响。若是 B 被击落，由于己方攻击的一直是 A。此时假设 B 攻击了 A k 步，A

攻击了 B j 步，则 A 的反馈信息 p_{Af} 为 $\prod_{i=1}^{k+n} \delta_i * p_{A1}$ ，则 B 的反馈信息 p_{Bf} 为

$\prod_{i=1}^j \delta_i * p_{B1}$ 。此时的权值 $E_a=1, E_b=3$ ，调用 $\text{Max}(E_a * p_{Af}, E_b * p_{Bf})|_{A:B}$ （第一局比赛

属于**正常型**)求得下次的攻击对象。若游戏进行到A的第一架飞机在B的第二架飞机之前掉落,则 $E_a=E_b=3$,再次调用 $Max(E_a * p_{Af}, E_b * p_{Bf})|_{A:B}$ 求出下次攻击的对象。若游戏中A,B双方有一人的第二架飞机被击落,则直接进入类似于问题一的模型,只需根据所有的反馈信息求出最大概率点进行攻击,直到本局比赛结束。

■若最初击打的是B,则与上述流程原理相同。

在A,B两方积分相等时,他们地位平等,可任意选择一方作为对手打击,因此第一局的开始击打对象是任意的。但游戏进行过程中由前若干局的积分的影响,我们需根据 $Max(Sum(A), Sum(B))|_{A:B}$ 求得下一局首先击打的对手。之后的游戏进行都与上述规律一致。

p_{Af} 和 p_{Bf} 与 δ_i , p_{A1} , p_{B1} 有关,而 p_{Af} , p_{Bf} 的求解与问题二的统计一样。

由以上的分析可以根据局数和比分的不同建立一下三个模型:

模型 III (i) ——正常型 (基本平分和落后的情况, 不包括最后几局)

与第一局比赛策略一致。因为无法影响其他两人的选择。因此只考虑自己所选择的最佳路径。首先通过 $Max(Sum(A), Sum(B))|_{A:B}$ 来选择这一局比赛开始时所选选择的最佳路径。然后同过问题二的模型选择该对象中的最大概率点(6, 4)。这一局比赛过程中,每一次选择通过 $Max(E_a * p_{Af}, E_b * p_{Bf})|_{A:B}$ 选择出每次攻击时所选择的对象。然后根据问题二中的模型由攻击A所得到的反馈信息可以得出这次攻击的最大概率点(i, j)。每次攻击之前根据反馈刷新 E_a 和 E_b 然后判断 $Max(E_a * p_{Af}, E_b * p_{Bf})|_{A:B}$,选择下一步的击打的对象,然后根据问题二中的模型由攻击A所得到的反馈信息可以得出这次攻击的最大概率点(i, j)。直到你退出则比赛结束或则其余一人退出。此时问题转换成问题二的模型,只剩下两方游戏。要做的就是根据问题二的策略选择打击的点对象,直到比赛结束。

模型 III (ii) ——保守型 (大比分领先其他两人的情况)

首先仍是通过 $Max(Sum(A), Sum(B))|_{A:B}$ 来选择这一局比赛开始时所选选择的击打对手,并且打击点为(6, 4)。由于自己大比分领先,在A,B双方分的权值不一致时,只要保证能拿分,而不必拿3分。比赛开始后每一步打通过 $Max(p_{Af}, p_{Bf})|_{A:B}$ 来选择击打对手。选择了击打对手后,然后根据问题二中的策略由攻击A所得到的反馈信息得出下一步打击的最大概率点(i, j)。下一次攻击根据反馈刷新 E_a 和 E_b ,继续循环,直到你退出则比赛结束或则其余一人退出。此时问题转换成问题二的模型,只剩下两方游戏。要做的就是根据问题二的策略选择打击的点对象,直到比赛结束。

模型 III (iii) ——博击型(最后几局比赛, 且己方分数落后的情况)

首先仍是通过 $\text{Max}(\text{Sum}(A), \text{Sum}(B))|_{A:B}$ 来选择这一局比赛开始时所选择击打的对手, 并且打击点为 (6, 4)。由于己方分数落后, 通过 $\text{Max}(E_a, E_b)|_{A:B}$ 来选择击打的对手, 以获得高分。确定了打击对手后根据问题二中的策略由已知的反馈信息决定打击的点为 (i, j)。之后根据反馈刷新 E_a 和 E_b , 重复循环, 直到你退出则比赛结束或其余一人退出。则转换成问题中 2 的模型, 只剩下 2 方游戏。此时问题转换成问题二的模型, 只剩下两方游戏。要做的就是根据问题二的策略选择打击的点对象, 直到比赛结束。

问题三通过 $\text{MAX}(\cdot)$ 确定打击的对手, 然后通过 3 种模型分类, 最终将问题转化为问题二的模型进行简化求解。

五. 模型的优缺点分析

在表格 (三) 中不难发现, 最终的所有可能出现的概率总和不等于 1, 所以说模型还有进一步的提高, 分析问题产生的原因。影响人打中机头的概率 P_1 的因数不仅只和步数有关, 是一个多方面影响的, 不是简单的线性关系, 既 P_1 不是一个常数。其次, 虽然在计算机模拟中可以发现计算机最多只要输入 8 次就可以, 但是在模型 2 中很难从数据中看出。模型没有考虑 8 次就一定结束, 只是计算了 8 次所有的概率和。

问题一和问题二的模型最大的优点是充分利用反馈信息形成对信息的筛选和删减, 根据反馈的信息进行下一步打击的目标。相比之下, 此种打击策略比固定打法的打击策略更有可行性和科学性。

问题三的是建立在人是随机选择的情况下。而如果三个同等智慧的人, 则应存在防守策略。在比分较为领先的情况下, 可能存在防守模型, 即你选择击打能使对方得到反馈信息最少的点。而问题三的模型的个优点在于可以充分利用模型二中的结论, 最大的缺点是没有考虑三个同智慧的情况下可能存在的防守模型。

六、参考文献

- [1] 杨振华、酆志新, 《数学实验》, 北京: 科学出版社, 2007。
- [2] 盛骥、谢式千、潘承毅, 《概率论与数理统计》(第三版), 北京: 出版社高等教育出版社, 2007。
- [3] 姜启源、谢金星、叶俊, 《数学模型》(第三版), 北京: 高等教育出版社, 2007。

附录一：

问题一源代码：

```
#include <iostream.h>
#define N 50
int count;
class plot{
public:
int pic[7][7];
int m,n;

plot &operator=(plot &);
};
plot &plot::operator=(plot &r)
{
    int i,j;
    if(this==&r)
        return *this;
    for(i=0;i<7;i++)
        for(j=0;j<7;j++)
            pic[i][j]=r.pic[i][j];
    m=r.m;
    n=r.n;
}
void pan(plot *p,int &i,int &j,int &n);
void find(int &m,int &n,int *c);
main()
{
    class plot array[N],brray[N];
    int i,j,a,b,num=0;
    int l,k;
    int c[7][7]={0};
    for(i=0;i<7;i++)
    {
        for(j=0;j<7;j++)
        {
            if(j+3<7&&i+2<7&&i-2>-1)//上
            {
                for(a=0;a<7;a++)
                {
                    for(b=0;b<7;b++)
                    {
                        if((b==j+3&&a>i-2&&a<i+2)||
                            (b==j+2&&a==i)||
                            (b==j+1&&a>i-3&&a<i+3)||
```

```

        (b==j&&a==i))
        array[num].pic[a][b]=1;
    else
        array[num].pic[a][b]=0;
    }
}
array[num].m=i;
array[num].n=j;
num++;
c[i][j]++;
}
if(j-3>-1&&i+2<7&&i-2>-1)//下
{
    for(a=0;a<7;a++)
    {
        for(b=0;b<7;b++)
        {
            if((b==j-3&&a>i-2&&a<i+2)||
                (b==j-2&&a==i)||
                (b==j-1&&a>i-3&&a<i+3)||
                (b==j&&a==i))
                array[num].pic[a][b]=1;
            else
                array[num].pic[a][b]=0;
        }
    }
    array[num].m=i;
    array[num].n=j;
    num++;
    c[i][j]++;
}
if(i+3<7&&j+2<7&&j-2>-1)//右
{
    for(a=0;a<7;a++)
    {
        for(b=0;b<7;b++)
        {
            if((a==i+3&&b>j-2&&b<j+2)||
                (a==i+2&&b==j)||
                (a==i+1&&b>j-3&&b<j+3)||
                (a==i&&b==j))
                array[num].pic[a][b]=1;
            else
                array[num].pic[a][b]=0;
        }
    }
}

```

```

        }
    }
    array[num].m=i;
    array[num].n=j;
    num++;
    c[i][j]++;
}
if(i-3>-1&&j+2<7&&j-2>-1)//左
{
    for(a=0;a<7;a++)
    {
        for(b=0;b<7;b++)
        {
            if((a==i-3&&b>j-2&&b<j+2)||
                (a==i-2&&b==j)||
                (a==i-1&&b>j-3&&b<j+3)||
                (a==i&&b==j))
                array[num].pic[a][b]=1;
            else
                array[num].pic[a][b]=0;
        }
    }
    array[num].m=i;
    array[num].n=j;
    num++;
    c[i][j]++;
}
}
}
for(i=0;i<7;i++)
{
    for(j=0;j<7;j++)
        cout<<c[i][j]<<' ' ;
    cout<<endl;
}
int max=c[0][0];
for(i=0;i<7;i++)
    for(j=0;j<7;j++)
        if(c[i][j]>max)
        {
            max=c[i][j];
            l=i;
            k=j;
        }
}

```

```

        cout<<l+1<<' ' <<7-k<<endl;
        pan(array, l, k, num);
    }
void pan(plot *b,int &e,int &f,int &num)
{
    int z;
    int c[7][7]={0};
    int p,q,cnt;
    int i,j;
    cout<<"输入是否击落，是输入 2，击中机身输入 1，否则为 0"<<endl;
    cin>>z;
    if(z==2)
    {
        cout<<"game over"<<endl;
    }
    else if(z==1)
    {
        cnt=0;
        for(int m=0;m<num;m++)
        {
            if(b[m].m!=e||b[m].n!=f)
            {
                {
                    if(b[m].pic[e][f]==1)
                    {
                        b[cnt]=b[m];
                        cnt++;
                        p=b[m].m;
                        q=b[m].n;
                        c[p][q]++;
                    }
                }
            }
        }
    }
    for(i=0;i<7;i++)
    {
        for(j=0;j<7;j++)
            cout<<c[i][j]<<' ';
        cout<<endl;
    }
    int max=c[0][0];
    for(i=0;i<7;i++)
        for(j=0;j<7;j++)
            if(c[i][j]>max)

```

```

        {
            max=c[i][j];
            p=i;
            q=j;
        }
        cout<<p+1<<' ' <<7-q<<endl;
        pan(b, p, q, cnt);
    }
    else if(z==0)
    {
        cnt=0;
        for(int m=0;m<num;m++)
        {
            if(b[m].pic[e][f]==0)
            {
                b[cnt]=b[m];
                cnt++;
                p=b[m].m;
                q=b[m].n;
                c[p][q]++;
            }
        }
        for(i=0;i<7;i++)
        {
            for(j=0;j<7;j++)
                cout<<c[i][j]<<' ';
            cout<<endl;
        }
        int max=c[0][0];
        for( i=0;i<7;i++)
            for(j=0;j<7;j++)
                if(c[i][j]>max)
                {
                    max=c[i][j];
                    p=i;
                    q=j;
                }
        cout<<p+1<<' ' <<7-q<<endl;
        pan(b, p, q, cnt);
    }
}

```

附录二：

问题二源代码：

```
#include <iostream.h>
#include <stdio.h>
#define N 500
int count;
class plot{
public:
int pic[9][9];
int m,n;
plot &operator=(plot &);
};
class plot2{
public:
int p1,p2;
plot2 &operator=(plot2 &);
};
plot &plot::operator=(plot &r)
{
    int i,j;
    if(this==&r)
        return *this;
    for(i=0;i<7;i++)
        for(j=0;j<7;j++)
            pic[i][j]=r.pic[i][j];
    m=r.m;
    n=r.n;
}
plot2 &plot2::operator=(plot2 &r)
{
    if(this==&r)
        return *this;
    p1=r.p1;
    p2=r.p2;
}
void pan(plot2 *p,plot * ,int &nu,int &j,int &n);
void pan2(plot *p,int &i,int &j,int &n);
main()
{
    class plot array[N];
    int i,j,a,b,d,num=0;
    int nu=0;
    plot2 br[5000];
    int pic[9][9]={0};
    int c[9][9]={0};
```

```

for(i=0;i<9;i++)
{
    for(j=0;j<9;j++)
    {
        if(j+3<9&&i+2<9&&i-2>-1)//上
        {
            for(a=0;a<9;a++)
            {
                for(b=0;b<9;b++)
                {
                    if((b==j+3&&a>i-2&&a<i+2)||
                       (b==j+2&&a==i)||
                       (b==j+1&&a>i-3&&a<i+3)||
                       (b==j&&a==i))
                        array[num].pic[a][b]=1;
                    else
                        array[num].pic[a][b]=0;
                }
            }
            array[num].m=i;
            array[num].n=j;
            c[i][j]++;
            num++;
        }
        if(j-3>-1&&i+2<9&&i-2>-1)//下
        {
            for(a=0;a<9;a++)
            {
                for(b=0;b<9;b++)
                {
                    if((b==j-3&&a>i-2&&a<i+2)||
                       (b==j-2&&a==i)||
                       (b==j-1&&a>i-3&&a<i+3)||
                       (b==j&&a==i))
                        array[num].pic[a][b]=1;
                    else
                        array[num].pic[a][b]=0;
                }
            }
            array[num].m=i;
            array[num].n=j;
            c[i][j]++;
            num++;
        }
    }
}

```

```

        if(i+3<9&&j+2<9&&j-2>-1)//右
        {
            for(a=0;a<9;a++)
            {
                for(b=0;b<9;b++)
                {
                    if((a==i+3&&b>j-2&&b<j+2)||
                       (a==i+2&&b==j)||
                       (a==i+1&&b>j-3&&b<j+3)||
                       (a==i&&b==j))
                        array[num].pic[a][b]=1;
                    else
                        array[num].pic[a][b]=0;
                }
            }
            array[num].m=i;
            array[num].n=j;
            c[i][j]++;
            num++;
        }
        if(i-3>-1&&j+2<9&&j-2>-1)//左
        {
            for(a=0;a<9;a++)
            {
                for(b=0;b<9;b++)
                {
                    if((a==i-3&&b>j-2&&b<j+2)||
                       (a==i-2&&b==j)||
                       (a==i-1&&b>j-3&&b<j+3)||
                       (a==i&&b==j))
                        array[num].pic[a][b]=1;
                    else
                        array[num].pic[a][b]=0;
                }
            }
            array[num].m=i;
            array[num].n=j;
            c[i][j]++;
            num++;
        }
    }
}
for(a=0;a<9;a++)

```

```
{
    for(b=0;b<9;b++)
        printf("%4d",c[b][a]);
    cout<<endl;
}
cout<<num<<endl;
for(i=0;i<num;i++)
{
    for(j=0;j<num;j++)
    {
        for(a=0;a<9;a++)
        {
            for(b=0;b<9;b++)
                pic[a][b]=array[i].pic[a][b]+array[j].pic[a][b];
        }
        int flag=0;
        for(a=0;a<9;a++)
        {
            for(b=0;b<9;b++)
            {
                if(pic[a][b]>flag)
                    flag=pic[a][b];
            }
        }
        if(flag==1)
        {
            br[nu].p1=i;
            br[nu].p2=j;
            nu++;
        }
    }
}
cout<<nu;
cout<<endl;
for(a=0;a<9;a++)
{
    for(b=0;b<9;b++)
    {
        pic[a][b]=0;
    }
}
for(i=0;i<nu;i++)
{
```

```

        a=br[i].p1;
        b=array[a].m;
        d=array[a].n;
        pic[b][d]++;
        a=br[i].p2;
        b=array[a].m;
        d=array[a].n;
        pic[b][d]++;
    }
    for(a=0;a<9;a++)
    {
        for(b=0;b<9;b++)
            printf("%5d",pic[a][b]);
        cout<<endl;
    }
    cout<<endl;
    int max=c[0][0];
    for(a=0;a<9;a++)
    {
        for(b=0;b<9;b++)
        {
            if(pic[a][b]>max)
            {
                max=pic[a][b];
                i=a;
                j=b;
            }
        }
    }
    cout<<i+1<<" "<<9-j<<endl;
    pan(br, array, nu, i, j);
    return 0;
}

void pan(plot2 *p,plot *array,int &nu,int &m,int &n)
{
    long int i;
    int z,j;
    int a,b;
    int m1,m2,n1,n2;
    int hed[9][9]={0};
    int cnt;
    int max;
    plot brr[N];
    cout<<"打机头输入 2，打机身输入 1，未击中输入 0";

```

```
cin>>z;
if(z==2)
{
    cnt=0;
    for(i=0;i<nu;i++)
    {
        a=p[i].p1;
        m1=array[a].m;
        n1=array[a].n;

        b=p[i].p2;
        m2=array[b].m;
        n2=array[b].n;
        if(m1==m&& n1==n)
        {
            brr[cnt]=array[b];
            p[cnt]=p[i];
            cnt++;
            hed[m2][n2]++;
        }
        if(m2==m&& n2==n)
        {
            brr[cnt]=array[a];
            p[cnt]=p[i];
            cnt++;
            hed[m1][n1]++;
        }
    }
    max=hed[0][0];
    for(i=0;i<9;i++)
    {
        for(j=0;j<9;j++)
        {
            printf("%4d",hed[i][j]);
            if(hed[i][j]>max)
            {
                max=hed[i][j];
                a=i;b=j;
            }
        }
    }
    cout<<endl;
}
```

```

        cout<<a+1<<" "<<9-b<<endl;
        pan2(brr, a, b, cnt);
    }
    else if(z==1)
    {
        cnt=0;
        for(i=0;i<nu;i++)
        {
            a=p[i].p1;
            m1=array[a].m;
            n1=array[a].n;
            b=p[i].p2;
            m2=array[b].m;
            n2=array[b].n;
            if((m1!=m || n1!=n)&&(m2!=m || n2!=n))
            {
                if(array[a].pic[m][n]==1 || array[b].pic[m][n]==1)
                {
                    p[cnt]=p[i];
                    cnt++;
                    hed[m1][n1]++;
                    hed[m2][n2]++;
                }
            }
        }
        max=hed[0][0];
        for(i=0;i<9;i++)
        {
            for(j=0;j<9;j++)
            {
                printf("%4d", hed[i][j]);
                if(hed[i][j]>max)
                {
                    max=hed[i][j];
                    a=i;b=j;
                }
            }
            cout<<endl;
        }
        cout<<a+1<<" "<<9-b<<endl;
        pan(p, array, cnt, a, b);
    }
    else if(z==0)

```



```

{
    cnt=0;
    for(i=0;i<nu;i++)
    {
        a=p[i].p1;
        m1=array[a].m;
        n1=array[a].n;

        b=p[i].p2;
        m2=array[b].m;
        n2=array[b].n;
        if(array[a].pic[m][n]==0&&array[b].pic[m][n]==0)
        {
            p[cnt]=p[i];
            cnt++;
            hed[m1][n1]++;
            hed[m2][n2]++;
        }
    }
    max=hed[0][0];
    for(i=0;i<9;i++)
    {
        for(j=0;j<9;j++)
        {
            printf("%4d",hed[i][j]);
            if(hed[i][j]>max)
            {
                max=hed[i][j];
                a=i;b=j;
            }
        }
        cout<<endl;
    }
    cout<<a+1<<" "<<9-b<<endl;
    pan(p, array, cnt, a, b);
}

void pan2(plot *b,int &e,int &f,int &num)
{
    int z;
    int c[9][9]={0};
    int p,q,cnt;
    int i,j;

```

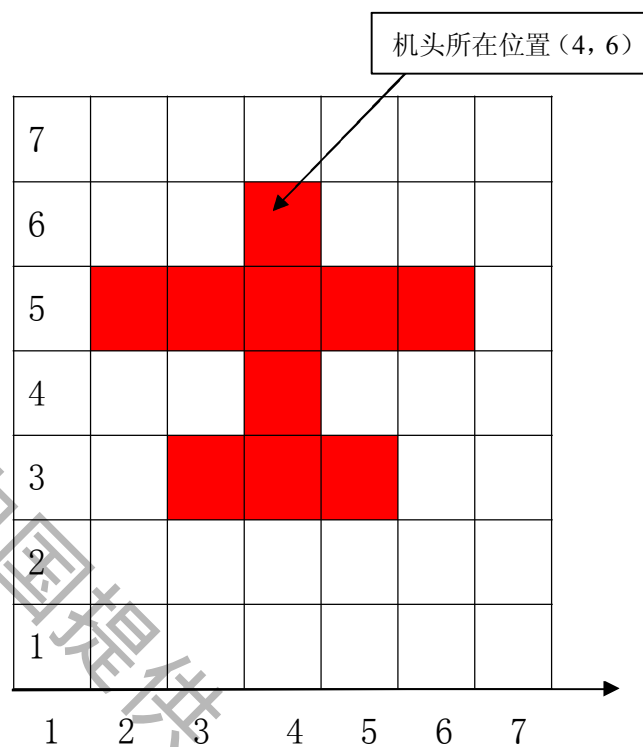
```

cout<<"输入是否击落，是输入 2，击中机身输入 1，否则为 0"<<endl;
cin>>z;
if(z==2)
{
    cout<<"game over"<<endl;
}
else if(z==1)
{
    cnt=0;
    for(int m=0;m<num;m++)
    {
        if(b[m].m!=e||b[m].n!=f)
        {
            if(b[m].pic[e][f]==1)
            {
                b[cnt]=b[m];
                cnt++;
                p=b[m].m;
                q=b[m].n;
                c[p][q]++;
            }
        }
    }
}
for(i=0;i<9;i++)
{
    for(j=0;j<9;j++)
        cout<<c[i][j]<<' ';
    cout<<endl;
}
int max=c[0][0];
for(i=0;i<9;i++)
    for(j=0;j<9;j++)
        if(c[i][j]>max)
        {
            max=c[i][j];
            p=i;
            q=j;
        }
cout<<p+1<<' '<<9-q<<endl;
pan2(b, p, q, cnt);
}
else if(z==0)

```

```
{
    cnt=0;
    for(int m=0;m<num;m++)
    {
        if(b[m].pic[e][f]==0)
        {
            b[cnt]=b[m];
            cnt++;
            p=b[m].m;
            q=b[m].n;
            c[p][q]++;
        }
    }
    for(i=0;i<9;i++)
    {
        for(j=0;j<9;j++)
            cout<<c[i][j]<<' ';
        cout<<endl;
    }
    int max=c[0][0];
    for( i=0;i<9;i++)
        for(j=0;j<9;j++)
            if(c[i][j]>max)
            {
                max=c[i][j];
                p=i;
                q=j;
            }
    cout<<p+1<<' ' <<9-q<<endl;
    pan2(b, p, q, cnt);
}
}
```

附录三（图表）



飞机示意图