商人与随从安全渡河综合决策运筹模型

摘要

本文针对商人与随从安全渡河综合决策运筹模型，使用图论理论、线性代数中的邻接矩阵技巧、动态规划和广度优先搜索算法，通过计算机模拟和以图论为基础的严格数学证明，建立了以多步骤决策模型为主的数学模型。 从确保商人与货物的安全角度出发，为其合理规划随从，安排每一步的渡河方法，提供了明确可行的方案。

为了研究商人与随从的决策选择，我们选定河流为一个理想环境。任何的天气因素、时间因素都不会对其商队决策和最终结果产生影响。

首先，我们依据计算机模型对商人人数为较小自然数的情况进行了模拟，利用广度优先搜索算法设计程序，得出了此情况下解的情况。之后，用朴素图论思想，利用商人仆人问题优良的可模型化性质提取模型，通过逆序法推出在商人人数为某一特定值时命题的真伪性。最后，以图论和邻接矩阵的计算，以严格的数学证明，给出了在商人人数为任意非零自然数时，命题的真伪性，并且提供了命题为真时的操作步骤。

经过检验和严格证明：在商人人数$m = 1, 2, 3$时命题为真，且可以给出具体且唯一的操作步骤。当$\mathrm{m} \geq 4$时，命题为假。

关键字 ：运筹学 决策模型 图论理论 邻接矩阵 动态规划 计算机模拟

一、 问题重述

1.1 问题背景

本题背景设定为四个商人携带商品和四个仆人，使用一条运载能力有限的船作为运输媒介，通过规划每一步的人员安排和行动策略，规避掉仆人人数对商人的制约和限制，成功地令队伍中八个成员渡过河流。以本题中四个商人和四个仆人的限定，在仆人与商人人数相等的前提下，我们自然猜想：当人数大于 4 时，命题是否有着相同的真伪性。

基于上述命题背景的具象化设定，我们可以抽象出其问题模型：多步决策问题。决策问题是指实际状态与期望状态之间存在的一种需要缩小或者排除的差距。决策问题是由实际状态、期望状态和差距三个要素组成的。实际状态首先是指现有的客观状态，有时也包括可预测、预计的未来状态。它有时是一些比较简单的事实现象，例如本题中抽象出来的商人与仆人携带物品过河的决策模型；有时则是一些比较复杂的事实现象总体，例如特斯拉自动驾驶汽车，使用决策模型来产生指令：加速，减速，变道等。因此，确认实际状态既要有实事求是的客观态度，又要掌握有效地了解客观事实的科学方法。 [1]

进而，我们可以将此问题归纳于运筹学的范围内。**运筹学**（Operations Research）,利用统计模型和数学模型等方法，去寻找复杂问题中的最佳或近似最佳的解答。运筹学是一门研究怎么样处理事情更有效的学科，比如机械动作合理安排，计算机的多线程，高层建筑材料的合理分配，不同动植物的共同养殖等都是当今社会经济发展的热点。[2]

1.2 问题要求

基于上述背景，我们需要建立数学模型解决以下问题：

1） 分析探究在本题目中设定条件下，建立数学模型探究问题是否有解（即命题的真伪性）。

2） 探究“四个商人和四个仆人”的基本情况是否有解，如果有解，则解答是否唯一，并尝试给出多个合理解答与步骤安排。

3） 在 1）2）的基础上，用数学方法严格证明命题在商人人数m为任意正整数时的真伪性，并且尝试给出具有普适性的解答方法。

二、问题分析

2.1 通过手动模拟方法探究解的存在性

通过观察可知，当$m=4$，情况较为简单。我们可以通过手动模拟的方法进行初步尝试和 探究，作为解决问题的初步探究。在m的值较小的情况下，我们可以大致通过猜测和手 动模拟，得出在一定程度上具有参考意义的结果，并且为计算机模拟的算法设计作出建设性指导。

2.2 通过计算机程序探究有限情况下解答的存在性

基于上问所建立的手动模拟结果，我们可以通过计算机进行初步的算法设计，并且进行模拟。通过数学分析，以动态规划为中心思想，配合广度优先搜索或者深度优先搜索，均可以得出一系列结果具有代表性、稳定、性质良好的结果——判定解的存在性。并记录下每次计算机程序给出的渡河策略，通过人工检验和模拟，确保计算机给出的渡河步骤确实能够似的商人和仆人顺利过河，保证实验的准确性。

2.3 通过数学严格证明 $m=\infty$ 时的命题真伪性

计算机程序通过大量计算，证明了m取不同值命题的真伪性。显而易见，计算机程序无法处理当$m=\infty$时的情况。我们以此为基础，进一步以数学方法严格证明在m取任意值时的存在条件。经过探究，以图论为基础，用邻接矩阵和有向图的方法，可以给出数学意义上对广义条件下本题的严格证明。

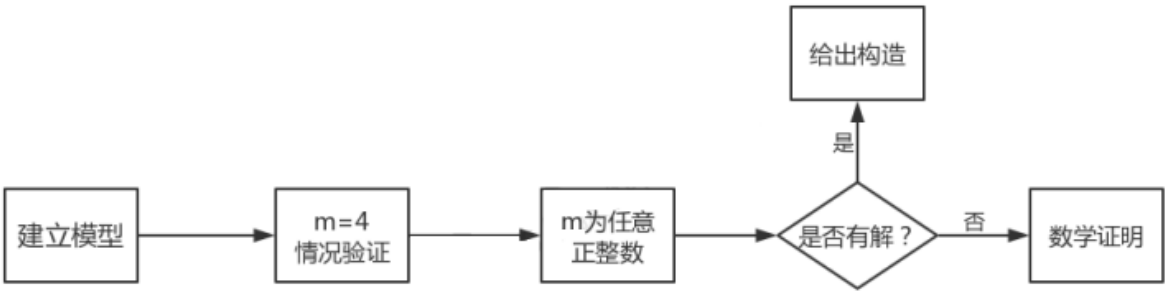

图 1.关于问题的探究与解决的流程

三、基本假设

为了便于探究和模型的建立，我们作如下假设：

1) 外界环境是理想环境，河流不会对船只渡河产生影响
2) 随从完全服从商人的命令，任意一个仆人可以和任意一个商人乘坐同一条船
3) 任意一个人都会划船，并且都能够单独驾驶船只渡河
4) 船只的数量是 1，且最大载客量为 2
5) 渡河是一个连续的过程，任何操作都需要时间来完成。
6) 仆人进行“杀人越货”的行为是几乎瞬时完成的。换言之，只要有一瞬间满足“杀人越货“的条件判定，仆人就会执行密约。商人不可能通过快速行动来避免条件判定的触发。

四、变量说明

符号	含义	单位
m	商人总数	人
l	船只最大运载量	人
r	此岸随从数量	人
b	此岸商人数量	人
S	安全状态集合	
d_k	第 k 个阶段做的决策	
x	横坐标	
y	纵坐标	
v	图的顶点	
O	零矩阵	

图 2.变量说明

五、模型的建立与求解

5.1.1 通过计算机程序构建的决策模型

利用计算机程序，我们首先对题目中所描述的状态进行了状态定义。
以二元集合(r,b)表示商队人物所处位置的状态，以仆从的密约“商人数如果小于仆从数，就杀人越货”为状态转移约束，建立了以广度优先搜索算法为理论基础，动态规划思想作为指导的模型，判定解的存在性，并且通过计算机模拟，给出有解情况下的每步操作。

5.1.2 “仆人“约束条件的求解

1） 术语使用的规定

将商队最初所处的岸边称为此岸，将其要去往目的地的岸边，称作为彼岸。

2） 引理的提出

定义状态：将二维向量 (r,b) 定义为状态。即在无人渡河时，此岸的人员分配情况，分别表示此岸存在 r 个随从，b 个商人。

对“状态”提出引理：状态(r,b)合法，当且仅当$r = b$, $b = 0$ 或 $b = m$

3）引理的严格证明

引理证明：

必要性：
当$r = b$时，商人安全。
当$b = 0$ 或 $b = m$ 时，一边岸上商人数为 0，与之相对的岸上商人数为 m，随从数小于等于 m，因此商人安全。

充分性：
假设：在状态$(\mathrm{r}_0,\mathrm{b}_0)$
下商人安全，且此时$\mathrm{r}_0 \neq \mathrm{b}_0$，$\mathrm{b}_0 \neq 0$，$\mathrm{b}_0 \neq \mathrm{m}$。
若一岸中存在$\mathrm{r}_0 < \mathrm{b}_0$，则在另一岸$\mathrm{m}-\mathrm{r}_0 > \mathrm{m}-\mathrm{b}_0$且$\mathrm{m}-\mathrm{b}_0 \neq 0$，其岸的仆人数量大于商人数量，因此商人会因仆人“密约”而被“杀人越货”。当$\mathrm{r}_0 > \mathrm{b}_0$时，同理。

5.1.3 状态转移方程的求解

由此岸渡河前往彼岸的过程中，状态(r,b)可以转移至状态$(\mathrm{r}-1,\mathrm{b}),(\mathrm{r}-2,\mathrm{b}),(\mathrm{r},\mathrm{b}-1),(\mathrm{r},\mathrm{b}-2),(\mathrm{r}-1,\mathrm{b}-1)$

由彼岸渡河返回此岸的过程中，状态(r,b)可以转移至状态$(\mathrm{r}+1,\mathrm{b}),(\mathrm{r}+2,\mathrm{b}),(\mathrm{r},\mathrm{b}+1),(\mathrm{r},\mathrm{b}+2),(\mathrm{r}+1,\mathrm{b}+1)$

将商人与仆人转化为二元集合(r,b)，自然地，我们可以将二元集合转化为二元坐标系下的点，建立二元坐标系图像。

由此岸渡河前往彼岸的过程中，如果可以从状态 X 到达状态 Y，则在二者之间以有向蓝线连结。

由彼岸渡河返回此岸的过程中，如果可以从状态 X 到达状态 Y，则在二者之间以有向红线连结。

由此，可得双色有向图，用以表示状态转移的全过程。

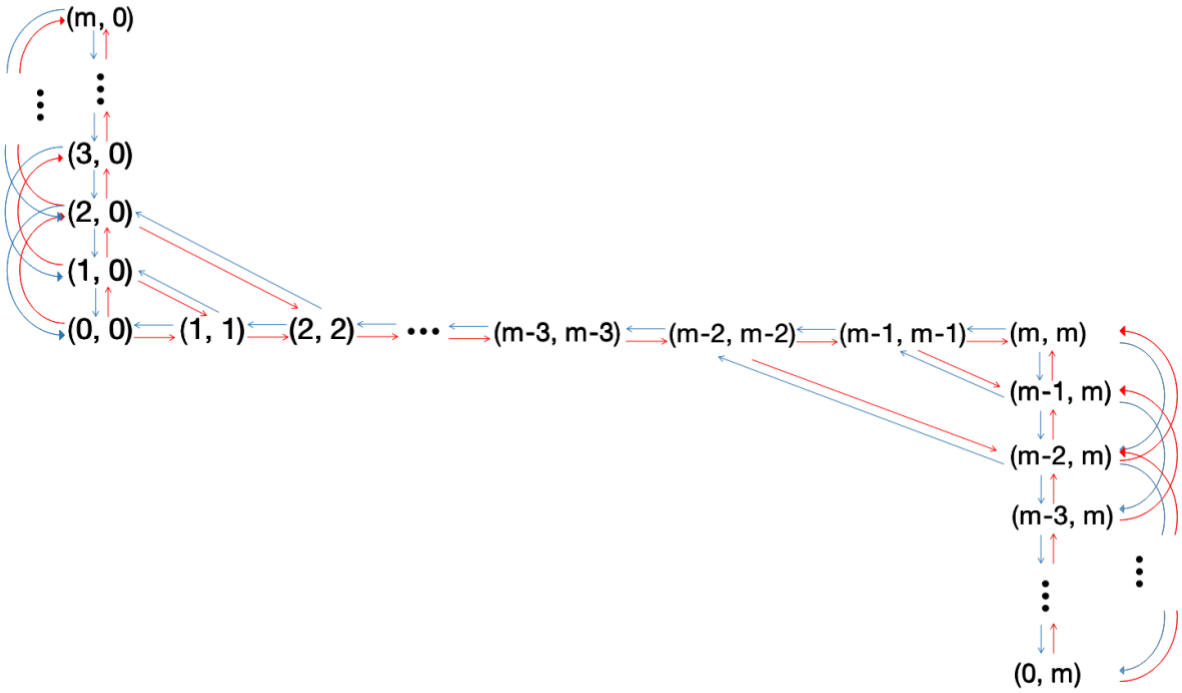

图 3.状态转移的全过程

5.1.4 问题有解性的判断

问题转换为：是否存在一条路径，使得每次交替经过红色和蓝色的边，可从(m, m)到达$(0,0)$？若存在，则此情况有解，反之无解。

5.1.5 商人数目$m = 4$时的计算机模拟情况

$m = 4$的图像如图所示

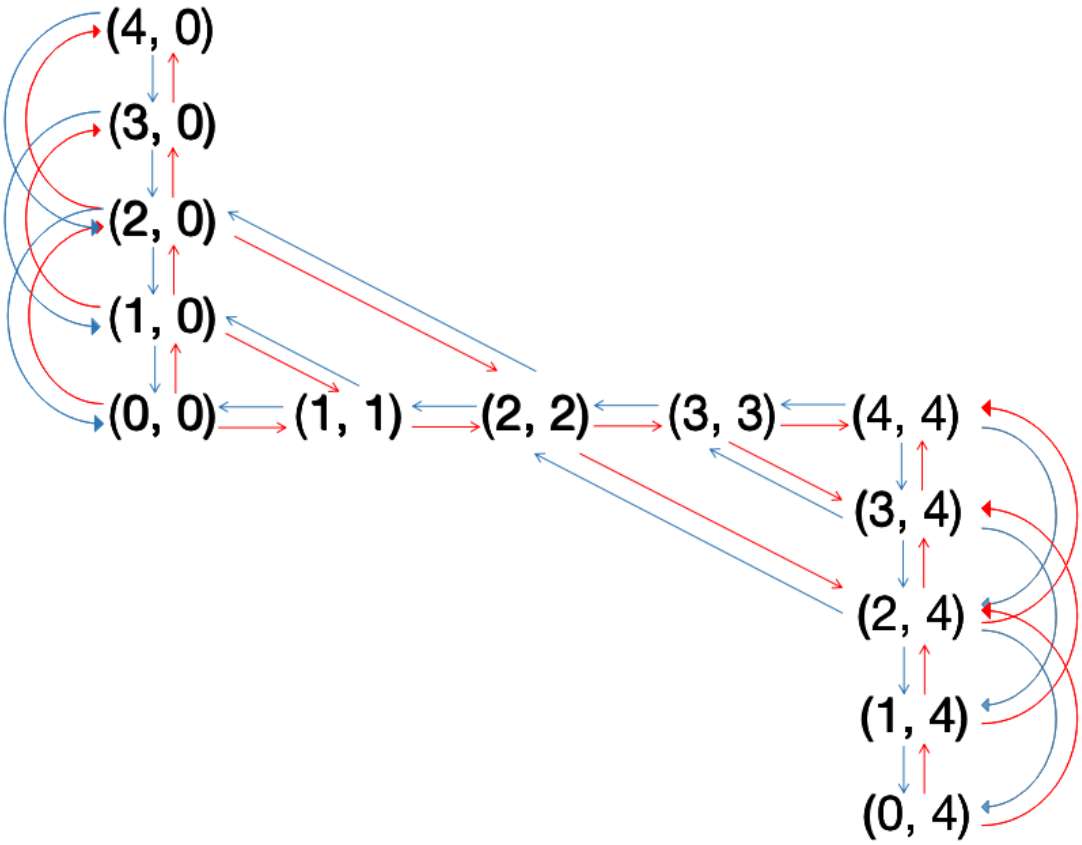

图 4. $m = 4$时的图像

从左上角到右下角的点分别赋值为 0，1，...，12，可以使用以下程序使用广度优先搜索证明无解

结果输出"No"，证明该图中不存在这样的一条路径。因此 4 个人的时候不存在一个这样的方案。

广度优先搜索的代码见附录 1

5.1.6 计算机模拟程序对 m 为有限任意正整数情况的讨论

我们给出$m = 1, 2, 3$时方案的构造，并且证明$m \geq 4$时不存在这样的方案

5.1.6.1 当$m = 1$时的情况

情况简单，只需要两个人一起过河即可

5.1.6.2 当$m = 2$时的情况

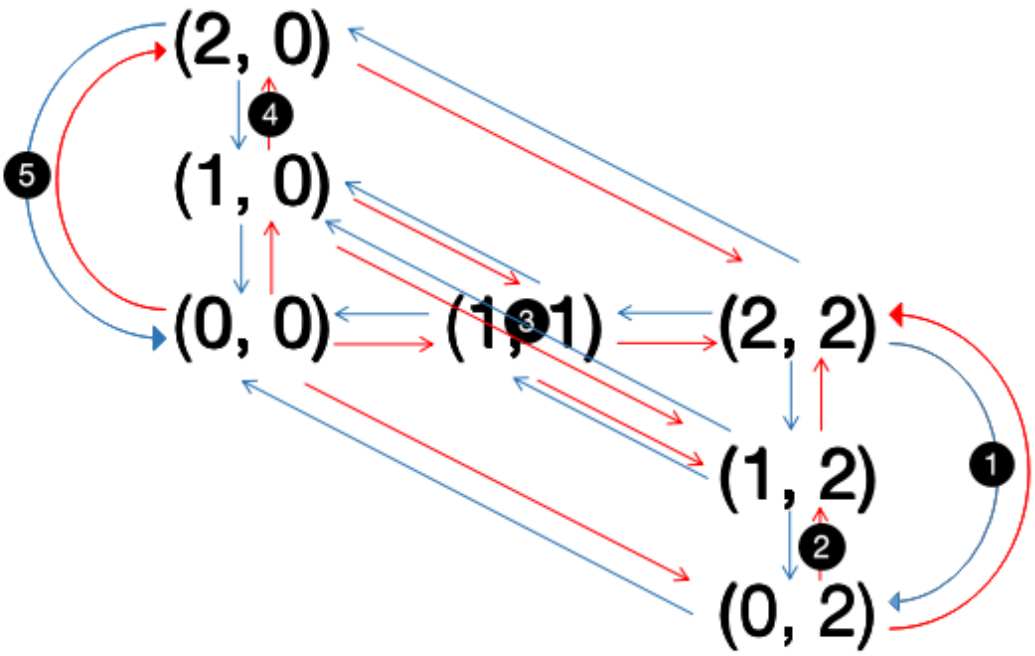

图 5. $m = 2$时的图像

如图所示。此过程中，我们从(2,2)状态，向目标状态(0,0)转移。

步骤解释：

1.两个随从乘船过河到达彼岸

2.彼岸的一个随从坐船回到此岸

3.此岸的两个商人坐船到达彼岸

4.彼岸的一个仆人坐船回道此岸

5.此岸的两个随从坐船到达彼岸

5.1.6.3 当m m = 3 时的情况

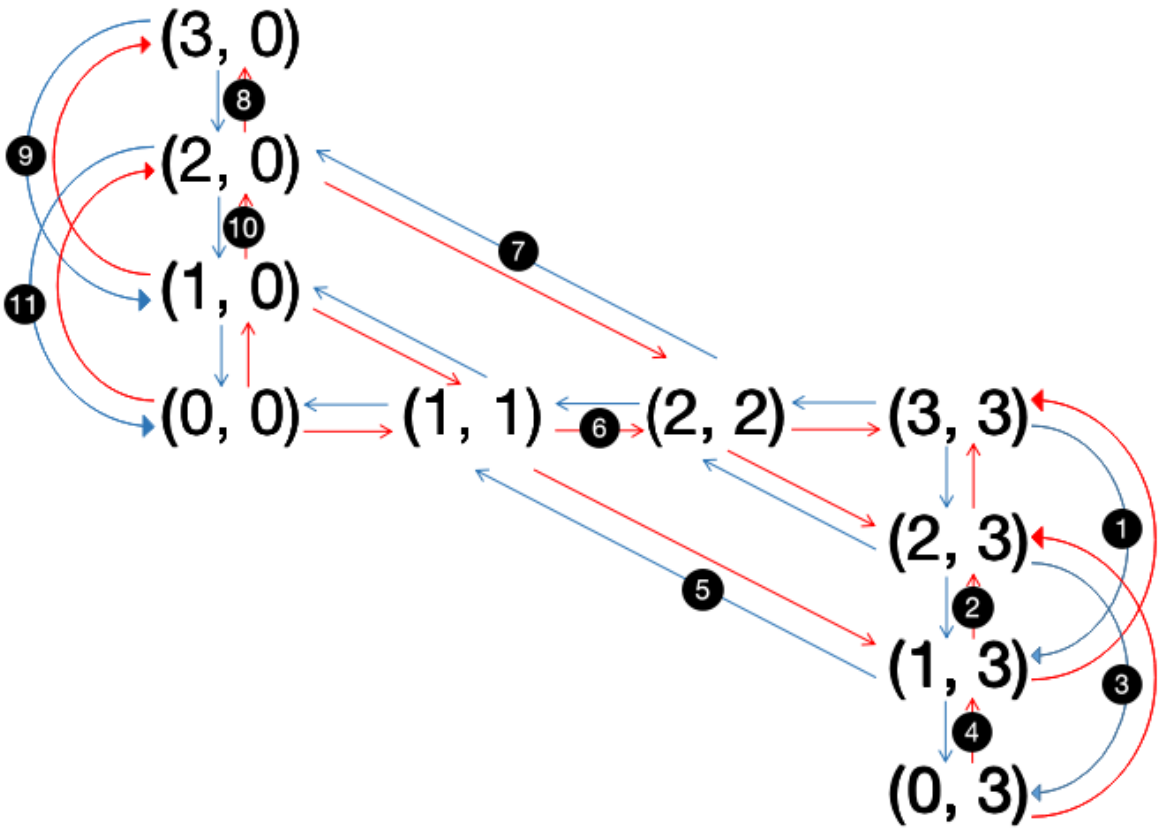

图 6. $m = 3$时的图像

如图所示。此过程中，我们从(3,3)状态，向目标状态(0,0)转移。

步骤解释：

1.此岸的两个随从乘船过河到达彼岸

2.彼岸的一个随从坐船回到此岸

3.此岸的两个随从乘船过河到达彼岸

4.彼岸的一个随从坐船回到此岸

5.此岸的两个商人坐船到达彼岸

6.彼岸的一个随从和一个商人坐船回到此岸

7.此岸的两个商人坐船到达彼岸

8.彼岸的一个随从坐船回到此岸

9.此岸的两个仆人坐船到达彼岸

10.彼岸的一个随从坐船回到此岸

11.此岸的两个仆人坐船到达彼岸

5.1.6.4 当 m = 4 时的情况

根据本文之前的讨论以及结果，此情况下命题为假命题。

5.1.6.5 证明 m >= 5 时命题的真伪性。

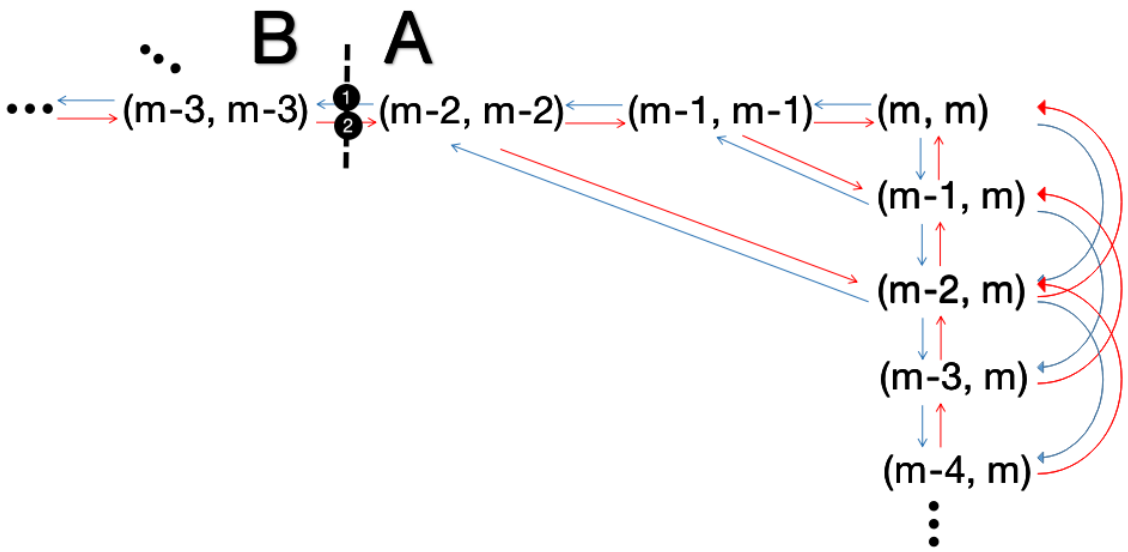

图 7. m >= 5 时的图像

证明：
(r, m)无法与连接，其中$b < m - 2$
$(r, 0)$无法与(r, b)连接，其中$b > 2$
(x, x)无法与$(x + k, x + k)$连接，其中$|k| > 1$
因此，如果去掉边①，②，整个图将被分为两个不连通的子图 A 和 B。要从 A 的点(m, m)到达 B 的$(0,0)$，必须经过$(m - 2, m - 2)$点，再从边①进入子图 B。但是在子图 A 中点$(m - 2, m - 2)$只有蓝色的入度，因此从 A 中的点进入$(m - 2, m - 2)$后不能进入蓝色的边①。因此无法从图 A 进入图 B。所以不存在一条路径从(m, m)到$(0,0)$且红蓝交替。

5.1.7 计算机模拟程序的评价

利用计算机程序，对题目中所描述的状态进行了状态定义。建立了以广度优先搜索算法为理论基础，动态规划思想作为指导的模型，判定解的存在性，并且通过计算机模拟，给出有解情况下的每步操作。此过程中，首先讨论了$m = 4$时的基本情况，以计算机程序判断无解。进而，从$m = 1$的情况开始依次讨论，基于图论的知识，给出了m等于任意正整数的解的存在性的情况，提供了有解情况$m = 1, 2, 3$ 时候行动步骤的具体操作。

六、模型的检验

我们通过计算机程序 取任意 m 属于正整数，与模型给出的结果均完美符合。说明了我们模型的准确性和完备性。

七、模型的评价

7.1 模型的优点

利用运筹学中的动态规划理论构建模型，然后使用广度优先搜索算法证明了 m 较小的情况，然后再用数学中的图论方法证明的 m 较大的情况，充分体现了多个学科交叉在数学建模中的运用。

7.2 模型的缺点

由于此模型是针对唯一解问题提出的，因此无讨论缺点的必要性。

7.3 模型的改进

我们提出了基于图论和邻接矩阵的建模优化

已知安全状态满足：

$$S = \{(r, b) | b = r, r = 1,2, \dots, m-1 \text{ 或 } b = 0, r = 0,1, \dots, m \text{ 或 } b = m, r = 0,1,, \dots, m\}$$

这样才能保证两岸商人都是安全的。

这些安全的状态共有$3m + 1$种。

设经过有限阶段可以将 m 个商人和 m 个随从从此岸转移到彼岸。s_k表示第 k 个阶段的状态用坐标表示，d_k表示第 k 个阶段做的决策
属于决策集$D = (1,0), (2,0), (1,1), (0,1), (0,2)$，即一个船上载多少商人、多少随从。

状态转移方程： $s_{k+1} = s_k + (-1)^{k-1} d_k$ 初始条件$s_0 = (0,0)$

当$m \geq 4$时，

将$x = 0, y = 0,1, \ldots, m$的点表示的顶点设为$v_1, v_2 \ldots v_{m+1}$
将$x = y, x = 1,2, \ldots, m-1$的点表示的顶点设为$v_{m+2}, v_{m+3}, \ldots v_{2m}$
将$x = m, y = 0,1,, \ldots, m$的点表示的顶点设为$v_{2m+1}, v_{2m+2}, \ldots v_{3m+1}$。

采用有向图的方法：

当船从此岸开到彼岸时，做出决策d_k，有向图可用图 8 表示，邻接矩阵为A；

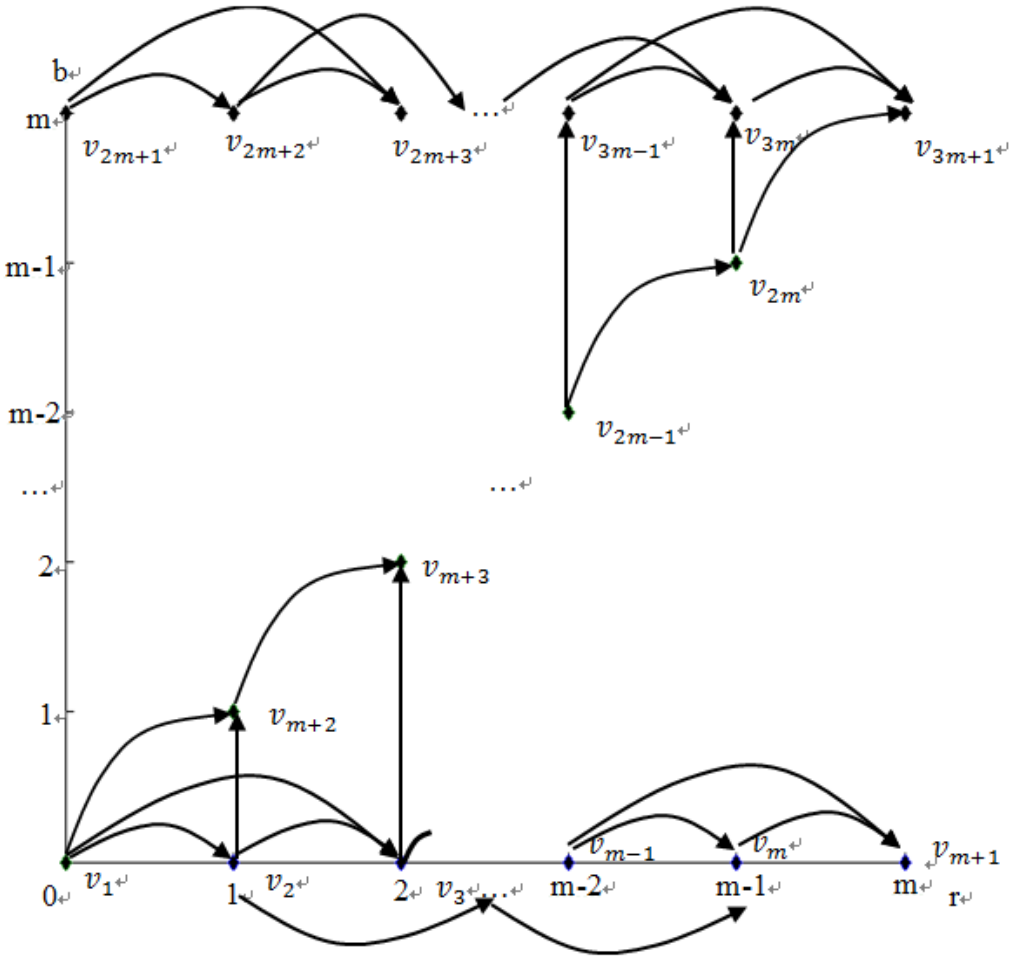

图 8. 决策示意一

当船从彼岸开到此岸时，做出决策$-d_k$，有向图用图 2 表示，邻接矩阵为B。

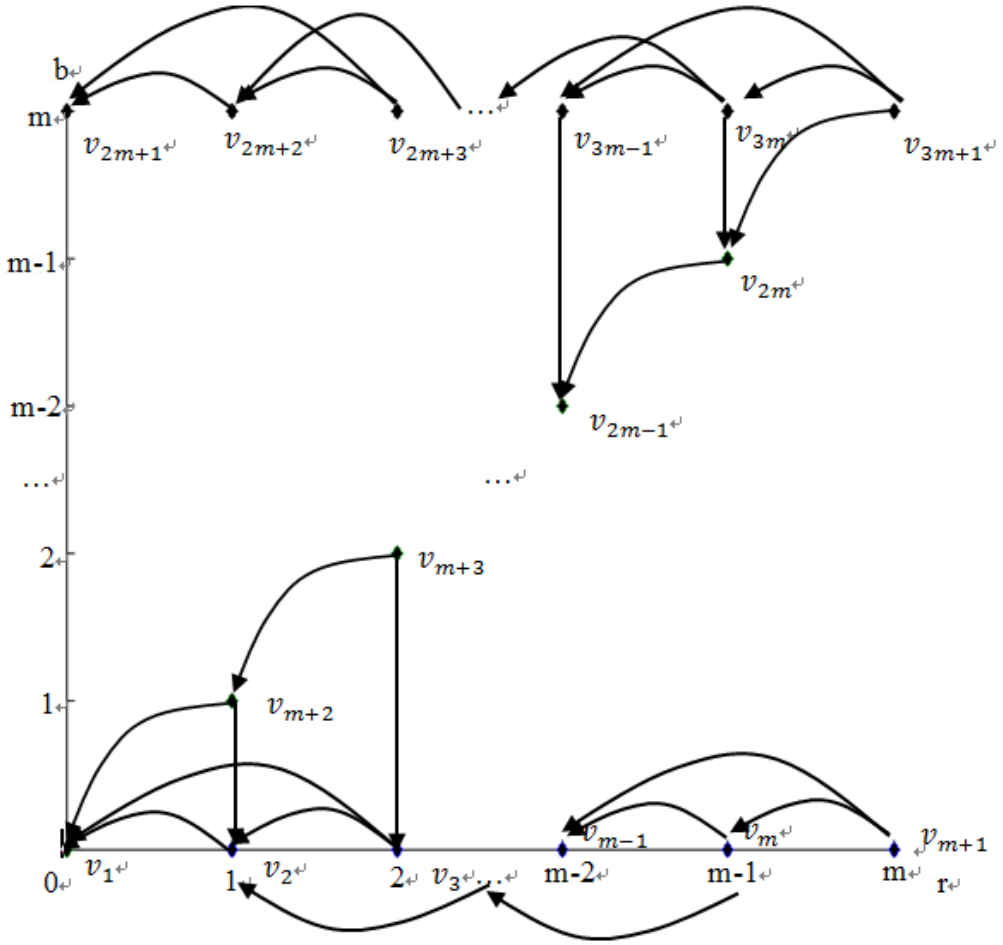

图 9. 决策示意二

两个过程先后发生，邻接矩阵为$\mathrm{C} = \mathrm{AB}$。

矩阵$(D = \mathrm{C^n A})$即可表示 n 次往返再过河后的情况

而一行人能从$(0,0)$经过有限次决策到达(m,m)，需要存在 n，使得$\mathrm{d}_{1,3\mathrm{m}+1} \neq 0$。

这种模型的优点在于更易于计算机操作。通过附录 2 中的 matlab 程序，计算机能使用此模型验证对于 m 的情况是否成立。

7.4 模型的推广

通过商人和仆人的模型，我们可以抽象出此类问题的一般情况：任意给定 n 种人，任意两种人之间可以存在制约关系。根据排列组合理论，可以存在 (n-1)! 种关系。同时

给定 m 种运载能力为 k 的运输工具，并且给定 p 个目的地。通过改变约束条件，和所要求的结果，可以有更多种题目的变化。

八、参考文献

[1] 萧浩辉． 决策科学辞典： 人民出版社， 1995

[2] wiki -运筹学

[3] Ewenwan’s Github Repository : Mathematics 经典模型

[4]数学模型（第四版）-姜启源 叶金星

[5]CSDN Blog 罗远航 ANC 3.0 License

附录 1: 基于广度优先搜索算法的计算机模拟代码

```
#include <iostream>
#include <vector>
#include <utility>
#include <queue>
#define pii pair<int, int>
using namespace std;

struct Node
{
	int id; int r, b;
}nodes[13];

struct Edge
{
	int u, v; int col;
}edges[44];

vector<int> G[13];

void build(int u, int v, int i)
{
	edges[2*i] = Edge{u, v, 0}; G[u].push_back(2*i);
	edges[2*i+1] = Edge{v, u, 1}; G[v].push_back(2*i+1);
}
```

```
bool bfs()
{
	int vis[13][2] = {0};	// 在 col 状态下是否已经到达过点 i
	queue<pii> q;
	q.push(make_pair(8, 1));
	vis[8][1] = 1;
	while(!q.empty())
	{
		pii p = q.front(); q.pop();
		int u = p.first, col = p.second;
		for(int i = 0; i < G[u].size(); i++)
		{
			Edge e = edges[G[u][i]];
			if(e.col == col || vis[e.v][e.col]) continue;
			if(e.v == 4) return true;
			vis[e.v][e.col] = 1;
			q.push(make_pair(e.v, e.col));
		}
	}
	return false;
}

int main()
{
	for(int i = 0; i < 5; i++)
		nodes[i] = Node{i, 4-i, 0};
	for(int i = 5; i < 8; i++)
		nodes[i] = Node{i, i-4, i-4};
	for(int i = 8; i < 13; i++)
		nodes[i] = Node{i, 12-i, 4};

	build(0, 1, 0); build(0, 2, 1);
	build(1, 2, 2); build(1, 3, 3);
	build(2, 3, 4); build(2, 4, 5);
	build(3, 4, 6); build(5, 4, 7);
	build(5, 3, 8); build(6, 5, 9);
	build(6, 2, 10); build(7, 6, 11);
	build(8, 7, 12); build(8, 9, 13);
	build(8, 10, 14); build(9, 10, 15);
	build(9, 7, 16); build(9, 11, 17);
	build(10, 11, 18); build(10, 12,19);
	build(10, 6, 20); build(11, 12, 21);

	if(bfs()) cout << "Yes" << endl;
	else cout << "No" << endl;
	return 0;
```

}
```

## 附录 2 在 m>=4 下对任意 m 命题为假的计算机验证（matlab 代码）

```
clear;
home;
p=0;%用于真假判断的量，p=0 代表命题一定为假，p=1 代表命题不一定为假。
for m=4:100;   %商人、随从数,本次验证当 m 取遍 4-100 的所有整数，命题均为假。
m>=4 下 m 可以随意更改。
    A=zeros(3*m+1);
    for i=1:m
        A(i,i+1)=1;
        A(2*m+i,2*m+i+1)=1;
    end
    for i=1:m-1
        A(i,i+2)=1;
        A(2*m+i,2*m+i+2)=1;
    end
    for i=m-2:2*m-1
        A(i,i+1)=1;
    end
    for i=2:3
        A(i,m+i)=1;
        A(i+2*m-3,i+3*m-3)=1;
    end
    A(1,m+2)=1;A(2*m,3*m+1)=1;
    B=rot90(A,2);
    C=A*B;
    for i=2*m:3*m+1
        for j=1:m-2
            if C(i,j)~=0||C(j,i)~=0
                p=1;%当对某个 m，方阵 C 目标区域存在一个元素不为 0，命题就不一
定为假。
            end
        end
    end
end
disp(p);%运行结果一定为 0，即代表在 m 所取范围内命题为假。
```
```