穿越沙漠

一、 摘要

本文通过建立模型为穿越沙漠的游戏玩家提供各种场景下的行动策略，具体包括以下内容：

问题一要求我们求解第一关和第二关的最佳策略。我们对于最佳策略的定义是在截止日期前到达终点并保留最多的资金。因此，我们利用起点→村庄、村庄→村庄、村庄→终点间的水和食物的消耗与补充的传动性，建立了目标函数为到达终点的资金最多的整数线性规划模型，对模型中的未知参数做定量分析与讨论，通过枚举经过的村庄次数k来得到不同的路线情况，针对相同的路线情况，利用贪心的思想找到其局部最优的时间分配，从而得到此条路线的全部未知参数。然后将参数代入整数线性规划模型，即可得到此条路径到达终点最多的资金数，以及在各个补给点即起点和村庄的补给情况。最后，比较不同路径的局部最优时间分配下的最后保留资金，找到最大值，此时间分配下的路径即为最优策略。通过枚举计算与贪心选择，我们计算得到第一关的最多资金为10470元，第二关的最多资金为12730元，具体结果见Result表格。

问题二要求我们求解一名玩家在仅知道当天的天气情况下的最佳策略。由于此时未来的天气不可知，因此我们对于最佳策略的定义是在截止日期前到达终点并保留较多的资金。**对于第三关**，我们首先通过计算得到玩家不应该去矿山挖矿，而是选择直接去往终点这一结论。因此问题转化为直接去往终点的路线的最优时间分配。对于晴天天气，玩家一定会跨域区域，由于没有沙暴天气，我们只需讨论今天为高温天气时玩家的决策即可。我们的处理方法是将天气情况视作一个马尔可夫过程，利用问题一的天气数据作为样本，计算出对应天气的条件概率，计算出最终资金的期望，来指导玩家游戏。最终策略为：行走路线为1→5→6→13或1→4→6→13且高温与晴朗天气玩家在任意一区域都不停留。**对于第四关**，我们首先为玩家制定一个路径最优策略。由于沙暴情况较少所以暂不考虑,将高温和晴朗天气状态都归约为一种天气状态P,其状态下的水和食物消耗为高温与晴朗的值按比例ρ组成。ρ是问题一天气情况中高温和晴朗出现的频率之比。然后再给出在贪心策略下的四种路径方案,并且根据沙暴天气的惩罚因子减去相应的代价之后做比较,选择出一条最优的路径即起点→村庄→矿山→村庄→矿山→终点,由于我们需要在30天之内到达终点且剩余资金最大化,所以需要根据当日的天气情况分别对在矿山和其他地区做讨论。具体决策见5.2.2.1。

问题三中增加了玩家的个数，玩家数量的增加使得玩家的收益减少、消耗增加，因此我们需要为玩家提供合适的策略使得剩余资金尽可能多。**对于第五关**，我们需要在玩家已知所有天气的情况下为其制定完整的行动方案，首先基于第三关的结论：理性玩家不走矿山，直接到终点，我们定义额外消耗，即玩家为避免冲突新选择方案与最佳方案的消耗差额，以额外消耗最小为目标建立规划方程，最终结论为：两名玩家分别选择1→4→6→13和1→5→6→13，其中一人前三天均行走，另外一人第一、三四天行走，第二天停留是最佳的行动方式。**针对第六关**，由于玩家当天行动结束可以知道其他玩家的方案与资源剩余量，因此玩家可以通过以上信息以及当天天气做出下一天的决策。当玩家通过信息判断其余玩家不与自己发生冲突时，其按第四关方案决策；若判断可能会与其他玩家发生冲突，通过构造收益函数建立三人博弈模型，得出结论为：预测将与其他玩家在道路冲突时该玩家应选择在当地停留；挖矿时预测将有玩家也来挖矿则应停止挖矿。

关键词　整数线性规划模型 贪心思想 马尔可夫过程 概率期望 博弈论

二、问题重述

2.1问题背景

在某游戏中，游戏玩家利用初始资金在起点处购置水、食物两种资源，随后穿越沙漠，途中可能会经过村庄补给资源或者在矿山停留通过挖矿补充资金，游戏要求玩家必须在规定的时间内到达终点，并且届时持有的资金要尽可能多。具体规则如下：

（1）资源消耗数量：原地停留一天消耗基础消耗量；行走一天消耗2倍的基础消耗量；在矿山挖矿一天消耗3倍的基础消耗量，同时获得基础收益（到达矿山的次日才可挖矿）；在矿山不挖矿消耗基础消耗量

（2）资源价格：起点处购买资源花费基础价格（限购买一次），在村庄购买资源花费2倍的基础价格，终点处按基础价格的一半退回资源

（3）资源持有量：玩家每天持有的资源质量不得超过负重上限，未到达终点前水和食物都不得耗尽

（4）天气：共有晴朗、高温、沙暴三种天气情况且所有区域天气状况相同，沙暴天气必须原地停留但是可以挖矿

（5）其他：该游戏以“天”为基本时间单位，游戏开始时间为第0天，玩家在规定时间内到达终点则游戏结束；玩家可原地停留也可行走，行走是只能到达所在区域的相邻区域

2.2问题提出

（一）在玩家数为1、全部天气已知的情况下，建立模型分析玩家的最优策略，在各参数给定的情况下求解“第一关”和“第二关”

（二）在玩家数为1、仅知道当天的天气并据此做出当天决策的情况下，给出玩家最优策略并求解“第三关”和“第四关”

（三）在有n名玩家的情况下，各玩家持有的初始资金相同且同时从起点出发，规则发生如下变动：

①任意$k\left(2\le k\le n\right)$名玩家在某一天均从区域A行走到区域B（$\mathrm{B}\neq\mathrm{A}$），各玩家的资源消耗数量均为基础消耗量的$2k$倍；

②任意$k\left(2\le k\le n\right)$名玩家某一天在同一矿山挖矿，各玩家的资源消耗数量均为基础消耗量的3倍，一天可通过挖矿获得的资金是基础收益的$\frac{1}{k}$；

③任意$k\left(2\le k\le n\right)$名玩家某一天在同一村庄购买资源，每箱价格均为基准价格的4倍。其他情况下消耗资源数量与资源价格与单人游戏相同

完成以下问题：

（1）在事先知道所有天气状况且各玩家在第0天确认行动方案并且之后不改动的情况下给出玩家的最优策略，并求解第五关

（2）若各玩家仅知道当天的天气状况，可以根据其余玩家当天的行动方案和自己持有的资源数量决定第二天的行动，给出玩家的最优策略并求解第六关

三、模型假设

（1）第一问的天气情况的频率能近似作为各个天气在沙漠中的概率
（2）玩家同步冲突带来的消耗比高温、沙暴等天气带来的额外消耗大
（3）第六问假设每个玩家都要追求最高利益

四、问题分析

问题一要求我们根据已有的每天天气状况，找到一般情况下玩家的最优策略。我们对于最优策略的理解是：玩家能够在规定时间内到达终点，并保留最多的资金。对于“第一关”和“第二关”，我们首先将平面图转化为其对偶图，考虑使用$Floyed$算法，求解出所有顶点对之间即所有区域之间的最短距离即最短时间。由于此时挖矿收入较高，因此我们利用贪心思想，以“尽快到达矿山”和“挖矿时间尽量长”等作为贪心原则，并结合实时天气，确定出玩家的从起点到终点的路线。随后，利用村庄可以补给的特性，我们可以求解出起点→村庄、村庄→村庄、村庄→终点间的水和食物的消耗，建立线性规划模型求解保留的最大资金，以及在起点或村庄需要购买的水和食物的数量，从而得到玩家完整的每一天的水和舒服的数量以及剩余资金的数目。

针对问题二，该问题要求我们在只有一名玩家且只知道当天天气的情况下给出玩家的最佳策略并求解第三关和第四关。对于一般情况的决策方案，我们的思路是先确定最优路线，再确定行动方案，具体来看，第三关没有涉及村庄，只涉及到了获取资金的矿山，因此玩家的行走路径主要有两类：经过矿山采矿并到达终点和不经过矿山直接到达终点，我们可以先从中选出最优路线再根据天气进行具体的决策；对于第四关，由于未来的天气是未知的，我们考虑利用第一问中的天气通过马尔可夫过程得到未来天气的状况，综合考虑晴朗和高温的消耗来确定最优路线，随之进一步确定具体的决策。

在问题三中，题目将玩家数量从1名增加至n名，并将规则修改为任意k名玩家在某天从A区域行走至B区域（A≠B）后行走消耗的资源数量增加，挖矿消耗的资源数量增加，挖矿带来的资金收益减少，因此玩家在原来基础上还要考虑其余玩家的行动方案并作出自己的决策。在第五关中，玩家数量为2名，每天的天气状况均为事先已知的，玩家需要是先做出完整的行动方案并且不能改动，在制定行动方案时，依据第三关中得到的玩家直接去终点、不经过矿山收益更高的结论，我们定义一个叫“额外消耗”的量，其表示的是玩家在走某条路径的时候为避免与另一名玩家同步而被惩罚选择利用时间错开时，比原路径最优的过关策略消耗多出的部分。我们列出了最小耗时为3的两条路径和最小耗时为4的两条路径，将其两两组合(允许相同)，考虑此时的额外消耗，从这些组合中找出额外消耗最小的那一组，即为在第0天给两名玩家的策略，其能满足总的额外消耗达到最少。换言之，剩下的资金就最多。针对第六关，未来天气位未知，仅能通过当天结束后他人的决策和剩余资源、当天天气来决策，我们分为两大类情况来讨论，第一类：玩家判断下一天不会与其他玩家发生冲突，此时决策参考第四关内容；第二类：玩家判断下一天会和其他玩家发生冲突，因此需要做出是否避让的博弈，我们打算通过博弈模型来解决这一问题。

五、模型建立与求解

5.1 问题一的模型及求解

5.1.1第一关、第二关数学模型建立

（一）符号说明

表1.1　符号说明表

符号	符号说明
w_{up}	负重上限
m_0	初始资金
t_{up}	截止日期
g	基础收益
w_0, w_1	水和食物的单位质量
p_0, p_1	基准价格
c_{00}, c_{01}	晴朗天气基础消耗量
c_{10}, c_{11}	高温天气基础消耗量
c_{20}, c_{21}	沙暴天气基础消耗量
D	最短距离即最短耗时矩阵
f	补给点间是否选择挖矿
Δt	挖矿时间即天数
$cost$	补给点之间的资源消耗
d	关键点的最短距离即最短耗时

（二）数据预处理

首先，我们将地图信息转化为图模型，即得到原地图平面图的对偶图，但需要剔除位于原地图外平面的那个在对偶图中的新顶点。由此，我们得到关卡一和二的图模型，设为$G = (\mathrm{V}, \mathrm{E})$，其为无向无权图。

随后，我们利用$floyd$算法，求得所有区域相互之间的最短距离矩阵D，由于$e_{ij} = \{0,1\}$，因此最小距离矩阵即代表最短耗时天数矩阵。利用矩阵D将关键节点起点、村庄、矿山和终点的相关最短耗时画出，如图1.1和图1.2所示：

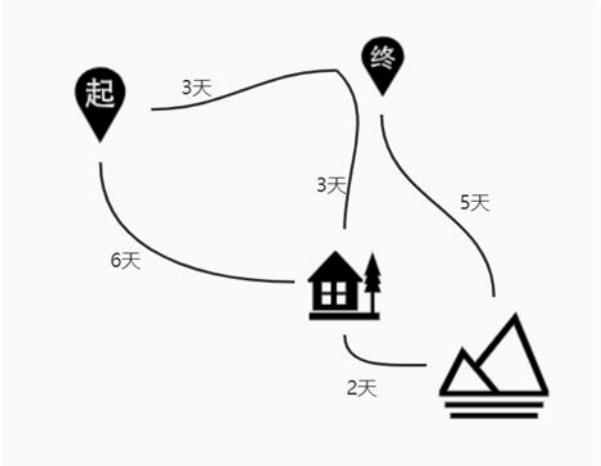


图1.1　第一关关键节点图

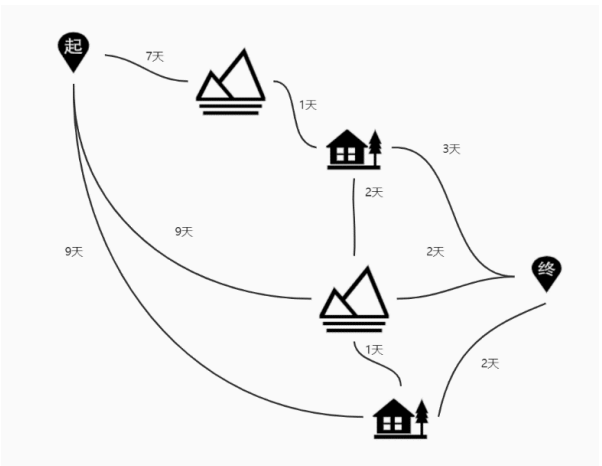


图1.2　第二关关键节点图

（三）模型建立与分析

设在从起点→终点的路程中，经过了k次村庄，画出示意图如图1.3所示

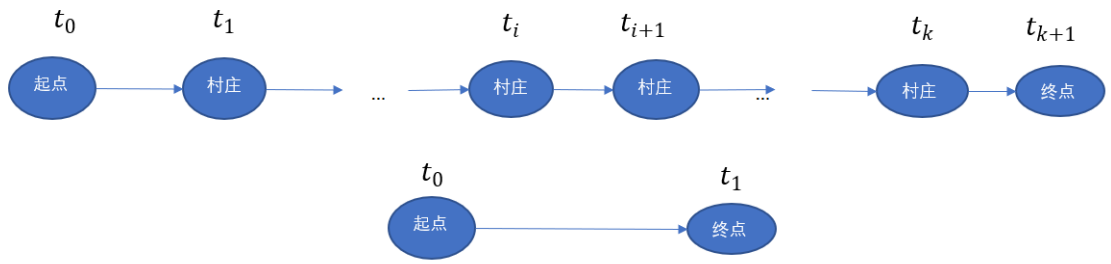


图1.3　起点→村庄→终点示意图

为了形成对比，我们将$k=0$时的情况单独画出，由地图信息可知，对于关卡一和二来说从起点→村庄的路途中通常会有矿区，起点→矿区通常会有村庄。因此在$k=0$的情况下，我们认为玩家不会经过矿区挖矿，也不会经过村庄补给。故此时即表示从起点直接到终点的情况。因此将其作为对照组，来检验进入矿区挖矿的路线是否产生了收益。

由图1.3可知，每个起点→终点的“箭头”代表一段时间，记玩家第i次到达村庄的时间为t_i，$i \in [1,k]$。同时记到达起点的时间为t_0，其值为0；到达终点的时间为t_{k+1}。在时间段(t_i,t_{i+1})中玩家可以选择去挖矿或不去，其中$i \in [0,k]$。设f_i表示第i个“箭头”即在时间段(t_i,t_{i+1})中是否选择挖矿，其中$f_i=1$表示选择挖矿，$f_i=0$表示不选择挖矿。设Δt_i表示时间段(t_i,t_{i+1})的挖矿时间，$cost_{i0},cost_{i_1}$表示时间段$(t_i,t_{i+1}]$的水和食物的消耗值，单位为箱。

下面将依据以上信息和变量建立保留最多资金的整数线性规划模型。

设在每个区域购买的资源即水和食物为x_i,y_i，其中$i \in [0,k]$。则玩家的资金变化来自以下五方面：

（1）初始的资金$m_0=10000$。

（2）玩家在起点购买资源消耗的资金m_1，即

$$m_1 = p_0x_0 + p_1y_0 \tag{1.1}$$

其中p_0,p_1分别表示水和食物的基准价格。

（3）玩家选择挖矿获得的收益m_2，即

$$m_2 = \sum_{i=0}^{k} f_i\, g\Delta t_i \tag{1.2}$$

其中g表示挖矿每天获得的基础收益，其值为1000元。

（4）玩家在中途村庄补给资源所消耗的资金m_3，此时购入价格为基准价格的2倍。即

$$m_3 = \sum_{i=1}^{k} 2\,p_0x_i + 2p_1y_i \tag{1.3}$$

（5）玩家到达终点后将剩余的资源退回的收益m_4，此时的退回价格为基准价格的一半，即

$$m_4 = \sum_{i=0}^{k} 0.5\,p_0(x_i - cost_{i0}) + 0.5p_1(y_i - cost_{i1}) \tag{1.4}$$

故游戏结束后，玩家持有的总资金m可以表示为式()

$$m = m_0 - m_1 + m_2 - m_3 + m_4 \tag{1.5}$$

下面考虑约束条件：

（1）负重约束：

①在起点不能购买超过负重上限w_{up}的资源，即

$$w_0 x_0 + w_1 y_0 \le w_{up} \tag{1.6}$$

②在每个补给点购买资源后，玩家所持有的资源不能超过负重上限w_{up}，即

$$w_0[\sum_{t=0}^{i-1}(x_i - cost_{i0}) + x_i] + w_1[\sum_{t=0}^{i-1}(y_i - cost_{i1}) + y_i] \le w_{up}, i \in [1,k] \tag{1.7}$$

（2）顺利过关的约束：

①需要在截止日期前(包括截止日期)到达终点，即

$$t_{k+1} \le t_{up} \tag{1.8}$$

②未到达终点前水和食物都不能耗尽，即在每次补给时水和食物箱数都需要大于等于0，因此有

$$\begin{cases} \sum_{i=0}^{j} x_i - cost_{i0} \ge 0 \\ \sum_{i=0}^{j} y_i - cost_{i_1} \ge 0 \end{cases}, j \in [0,k] \tag{1.9}$$

（3）资金约束：

①在起点不能购买超过初始资金m_0的资源，即

$$p_0 x_0 + p_1 y_0 \le m_0 \tag{1.10}$$

②每个补给点购买资源时不能购买超过当前资金的资源，即

$$\begin{aligned} &2p_0 x_i + 2p_1 y_i \le \\ &m_0 - (p_0 x_0 + p_1 y_0) - \sum_{t=2}^{i}(2p_0 x_{t-1} + 2p_1 y_{t-1} - f_{t-1} g \Delta t_{t_1}) \end{aligned} \tag{1.11}$$

其中$i \in [2,k]$。

③将第一个补给点单列出来，因为起点与村庄的售价不一样。即

$$2p_0 x_1 + 2p_1 y_1 \le m_0 - (p_0 x_0 + p_1 y_0) + f_0 g \Delta t_0 \tag{1.12}$$

因此，我们可以得到如下的线性规划模型：

$$\max m = m_0 - m_1 + m_2 - m_3 + m_4 \tag{1.13}$$

$$
s.t.\begin{cases}
\begin{cases} w_0x_0 + w_1y_0 \le w_{up} \\ w_0[\sum_{t=0}^{i-1}(x_i - cost_{i0}) + x_i] + w_1[\sum_{t=0}^{i-1}(y_i - cost_{i1}) + y_i] \le w_{up}, i \in [1,k] \end{cases} \\
\begin{cases} t_{k+1} \le t_{up} \\ \begin{cases} \sum_{i=0}^{j} x_i - cost_{i0} \ge 0 \\ \sum_{i=0}^{j} y_i - cost_{i_1} \ge 0 \end{cases} ,j \in [0,k] \end{cases} \\
\begin{cases} p_0x_0 + p_1y_0 \le m_0 \\ 2p_0x_i + 2p_1y_i \le m_0 - (p_0x_0 + p_1y_0) - \sum_{t=2}^{i}(2p_0x_{t-1} + 2p_1y_{t-1} - f_{t-1}g\Delta t_{t_1}), i \in [2,k] \\ 2p_0x_1 + 2p_1y_1 \le m_0 - (p_0x_0 + p_1y_0) + f_0g\Delta t_0 \end{cases} \\
m_0 = 10000 \\
m_1 = p_0x_0 + p_1y_0 \\
m_2 = \sum_{i=0}^{k} f_i\, g\Delta t_i \\
m_3 = \sum_{i=1}^{k} 2\,p_0x_i + 2p_1y_i \\
m_4 = \sum_{i=0}^{k} 0.5\,p_0(x_i - cost_{i0}) + 0.5p_1(y_i - cost_{i1})
\end{cases}
$$

显然，此模型并不能直接求解x和y，其含有五组未知数：每一段时间资源的消耗值$cost$；每一次挖矿的时长Δt；经过的村庄次数k；是否选择挖矿的变量f；到达每一个补给点的时间t。我们在此以玩家在非沙暴天气只在矿山可能停留为例，来列出这五组变量的关系：

我们引入几个关键区域的最短距离即最短耗时，记为d，其定义如下：d_1为起点→村庄的最短耗时；d_2为起点→矿山的最短耗时；d_3为矿山与村庄间的最短耗时；d_4为村庄→终点的最短耗时；d_5为矿山→终点的最短耗时。这些值都是在矩阵D中已知的。

（1）到达补给点的时间与挖矿的关系

①起点→第一个村庄：

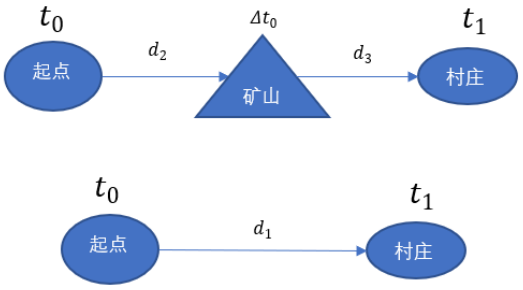


图1.4　起点→第一个村庄时间示意图

若是选择挖矿即$f_0 = 1$，则有

$$t_0 + d_2 + \Delta t_0 + d_3 = t_1 \tag{1.14}$$

若是不选择挖矿即$f_0 = 0$, 则有

$$t_0 + d_1 = t_1 \tag{1.15}$$

②村庄与村庄之间，此时一定会选择挖矿：

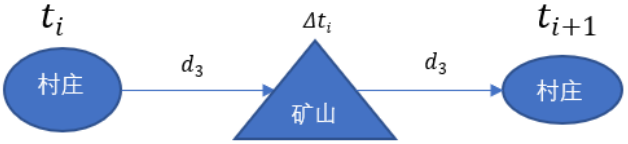
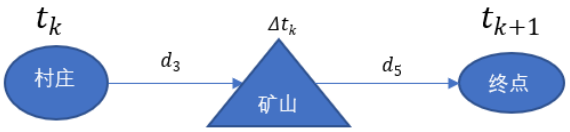


图1.5　村庄与村庄之间时间示意图

有下式：

$$\begin{cases} t_i + d_3 + \Delta t_i + d_3 = t_{i+1},\ i \in [1, k-1] \\ f_i = 1 \end{cases} \tag{1.16}$$

③最后一个村庄→终点：



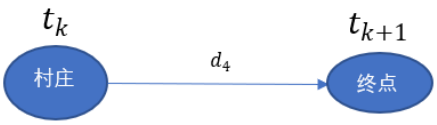


图1.6　最后一个村庄→终点时间示意图

若是选择挖矿即$f_k = 1$，则有

$$t_k + d_3 + \Delta t_k + d_5 = t_{k+1} \tag{1.17}$$

若是不选择挖矿即$f_k = 0$，则有

$$t_k + d_4 = t_{k+1} \tag{1.18}$$

此时，我们就可以得到t与$\Delta t, f$的关系。需要注意的是，在d的时间段内，每遇上一次沙暴天气，d的值需要增加一，因为沙暴天气无法跨越区域，只能原地停留。

（2）时间段内资源消耗与挖矿的关系

设函数$fwalk(t_0, t_1) = (t_1, cost_0, cost_1)$表示在非挖矿情况下时间段$(t_0, t_1]$的状态情况，其第一个值$t_1$表示实际结束的时间，因为当遇到沙暴天气时，必须停留在当前区域，因此结束时间t_1可能会向后移；第二、三个值表示这段时间消耗的水和食物的箱数。为方便表示，设其返回值为$cost_0, cost_1$。

其逻辑可以设置成如下形式：

$$cost_0 = 0, cost_1 = 0 \tag{1.19}$$

$$while(t0 \neq t_1)$$

$$do\begin{cases}\begin{cases}cost_0 = cost_0 + 2c_{00}\\cost_1 = cost_1 + 2c_{01}\end{cases}, if(weather_{t_0+1}) == 晴朗\\\begin{cases}cost_0 = cost_0 + 2c_{10}\\cost_1 = cost_1 + 2c_{11}\end{cases}, if(weather_{t_0+1}) == 高温\\\begin{cases}cost_0 = cost_0 + c_{20}\\cost_1 = cost_1 + c_{21}\\t_1 = t_1 + 1\end{cases}, if(weather_{t_0+1}) == 沙暴\\t_0 = t_0 + 1\end{cases}$$

$$return\ cost_0, cost_1$$

其中c_{i0}分别表示在天气为晴朗、高温、沙暴时的水和食物基础消耗量，$weather_i$表示第i天的天气。

类似地，我们可以设计一个$fmine(t_0, t_1)$的函数，记录挖矿情况下的状态情况，其逻辑如下形式：

$$cost_0 = 0, cost_1 = 0 \tag{1.20}$$

$$while(t0 \neq t_1)$$

$$do\begin{cases}\begin{cases}cost_0 = cost_0 + 3c_{00}\\cost_1 = cost_1 + 3c_{01}\end{cases}, if(weather_{t_0+1}) == 晴朗\\\begin{cases}cost_0 = cost_0 + 3c_{10}\\cost_1 = cost_1 + 3c_{11}\end{cases}, if(weather_{t_0+1}) == 高温\\\begin{cases}cost_0 = cost_0 + 3c_{20}\\cost_1 = cost_1 + 3c_{21}\end{cases}, if(weather_{t_0+1}) == 沙暴\\t_0 = t_0 + 1\end{cases}$$

$$return\ cost_0, cost_1$$

于是，我们考虑每一个时间段的资源消耗与挖矿的关系：

①时间段(t_0, t_1)有：

若是选择挖矿即$f_k = 1$，则有

$$\begin{aligned}\left(cost_{00}, cost_{01}\right) = fwalk\left(0, d_2\right)\\+fmine\left(d_2, d_2 + \Delta t_0\right)\\+fwalk\left(d2 + \Delta t_0, d2 + \Delta t_0 + d_3\right)\end{aligned} \tag{1.21}$$

若是不选择挖矿即$f_k = 0$，则有

$$(cost_{00}, cost_{01}) = fwalk(0, d_1) \tag{1.22}$$

②时间段$(t_i, t_i + 1), i \in [1, k-1]$有：

$$\begin{aligned}\left(cost_{i0}, cost_{i1}\right) = fwalk\left(t_i, t_i + d_3\right)\\+fmine\left(t_i + d_3, t_i + d_3 + \Delta t_i\right)\end{aligned} \tag{1.23}$$

$$+fwalk\left(t_i+d_3+\Delta t_i, t_i+d_3+\Delta t_i+d_3\right)$$

③时间段(t_k, t_{k+1})有：

若是选择挖矿即$f_k=1$，则有

$$\begin{aligned}\left(cost_{k0}, cost_{k1}\right)&=fwalk\left(t_k, t_k+d_3\right)\\&+fmine\left(t_k+d_3, t_k+d_3+\Delta t_k\right)\\&+fwalk(t_k+d_3+\Delta t_k, t_k+d_3+\Delta t_k+d_5)\end{aligned}\tag{1.24}$$

若是不选择挖矿即$f_k=0$，则有

$$(cost_{k0}, cost_{k1})=fwalk(t_k, t_k+d_4))\tag{1.25}$$

此时，我们就可以得到$cost$与$\Delta t, f, t$的关系。

综合以上两点，我们可以得到除k之外所有未知变量的关系。若k为已知，则可以确定$f, \Delta t$的可能情况，从而将所有未知变量解出，代入式（1.12）的线性规划模型即可解得x, y。而k在此题中并不大，因此，我们可以通过枚举k来实现求解的目的。

对于不同关卡，策略有一定的不同。针对第一关，第一关仅有一组矿山和村庄，即$(12,15)$，我们只需根据天气计算得到第一关的相关数据，并带入上述模型，即可求解。针对第二关，其有两组矿山和村庄，即$(30,39),(55,62)$。为此，我们需要额外考虑这两组矿山和村庄的组合，来求解计算相关数据，最后才能代入上述模型求解。为方便求解，我们仅考虑以下四种基本组合：

①起点→第一组，并在第一组一直重复，最后去到终点。

②起点→第一组，一段时间后，前往第二组，并最终去到终点。

③起点→第二组，并在第二组一直重复，最后去到终点。

④起点→第二组，一段时间后，前往第一组，并最终去到终点。

5.1.2模型求解

建立了上述模型后，我们对于一个确定的行走路线就变得可解。但是同一个行走路线，由于每天的天气不同，因此时间的安排会影响其最后剩下的资金数。进而求解原问题的的困难就不再是行走路线规划，而转换为了如何针对题目给出的已知的特定的天气状况，针对特定的行走路线，做出最优的时间安排。为此，我们采用了贪心[1]的思想，来更快地寻找较好的时间安排。

贪心思想一：一定程度上，挖矿天数越多，单次挖矿的天数越多，最后剩下的资金会越多

我们从挖矿一天的净收入来说明这一点，考虑从村庄那购买的双倍价格的水和食物，考虑消耗最大的沙暴天气，消耗的水和食物的总价值为

$$2p_0*3c_{20}+2p_1*3c_{21}=900$$

即在最坏的情况下，挖矿一天也能够净赚1000-900=100元。因此，我们在安排时间的时候，尽量让总的挖矿天数大。同时，我们不建议单次挖矿时间较短，因为到达矿山是有时间和资源成本的。特别是对于可能的矿山村庄来回移动的情况，如果某次挖矿时间比较短，那可能不会产生很好的收益。

贪心思想二：高温不停留的收益会更大

首先，高温与晴朗天气一天的基准消耗价格分别为8*5+6*10=100和5*5+7*10=95，非常地接近，即表示停留一天获得的收益并不客观。同时有了思想一，玩家需要尽快

赶到矿山来赚取收益，因此高温不停留。

贪心思想三：每次特意地补给资源时，玩家自身携带的资源数尽量少

若玩家本身资源数还较多，那么特意地去村庄补给资源地收益不高，因为去村庄以及可能再从村庄回来都含有路费成本。换句话说，不到非不得已的时候，不考虑特意地补给资源。

贪心思想四：到达终点时剩余的资源尽量为0

因为多买资源在到达终点后是亏钱卖出去的。若到达终点有多余的资源，则说明我们购买了本来不需要的资源，这是不合理的。因此我们应该选择在达到终点的过程中少买，刚好买够是最好的。这个规则可以在求解线性规划时通过改变补给来被强行执行，因此到达终点是不会有剩余的。

贪心思想五：尽可能在晴朗的天气下挖矿收益更高

因为在晴朗天气的时候挖矿食物和水消耗量最小消耗,相比于其他天气能赚取更多金钱。

基于以上规则，我们可以更快地寻找较好的时间安排。

5.1.2.1第一关分析与求解

首先，根据关卡一地图，我们发现从起点开始，村庄是矿山的必经路。因此f_0一定是0，即起点和第一次到达村庄之间不会去挖矿，因此前六天用于从起点走到村庄，因为有沙暴的天气，故实际花费的天数是8天。基于此，我们枚举经过村庄次数k，来寻找最优路线。

考虑$k=0$，此为对照组，即从起点直接去往终点，需要4天时间，由于第四天恰好沙暴，因此需要5天时间才能到达。我们并不需要着急计算出$k=0$时的最后资金，因为其一定小于m_0，如果$k\neq 0$时，最后的资金大于等于m_0，则自动舍弃掉此种情况。

考虑$k=1$，即如图1.7所示，有两种情况：

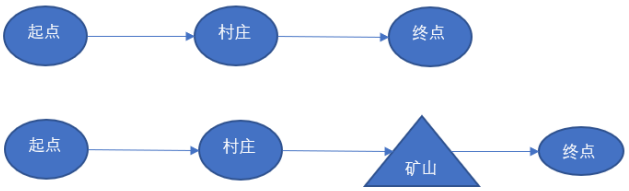


图1.7　k=1时示意图

显然第一种是不合理的，故舍去。

考虑$k=2$，即如图1.8所示，有两种情况：

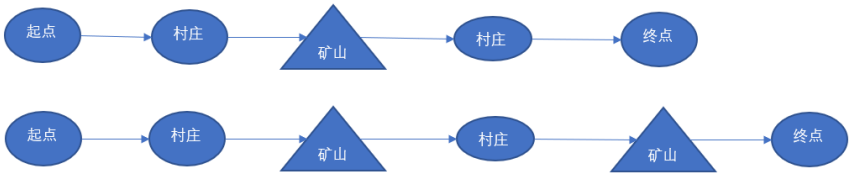


图1.8 k=2时示意图

考虑$k=3$，如图1.9所示，有两种情况：

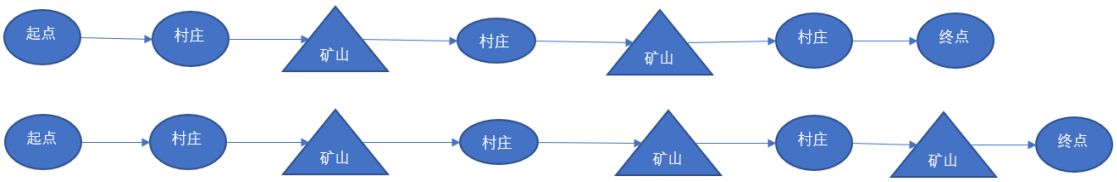


图1.9　k=3时示意图

我们此时来考虑第二种情况的可用于挖矿的总天数，最多为30-(8+2+2+2+2+3)=11天，而有三次挖矿，因此平均每次挖矿仅3.67天，而往返一次村庄和矿山需要四天，因此这种做法的收益绝对不会为最优。故当$k>3$时，我们不必考虑其可能的情况。综上分析，一共有4条可能的路线情况。利用之前的贪心思想以及关卡一的特定天气情况，我们得到了这四种路线的局部最优时间分配，如图所示1.10所示：

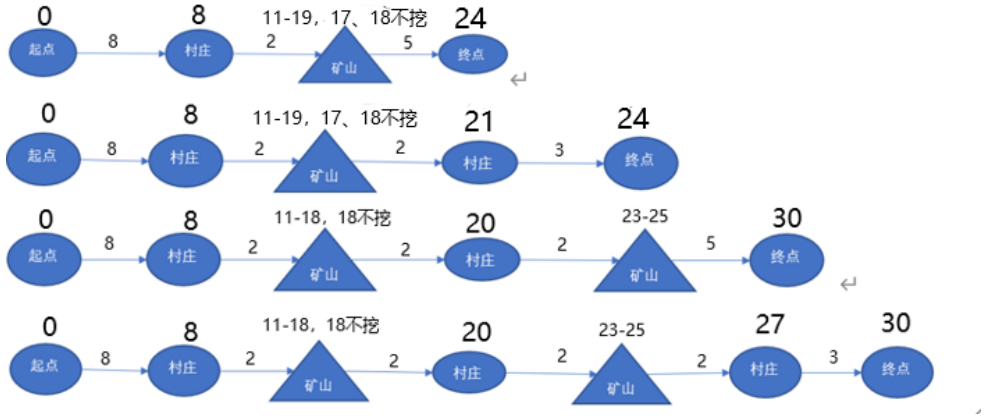


图1.10　四种可能路线局部最优时间分配

由上图所示线路1和线路二可以几乎等价看待，线路3和线路四可以几乎等价看待，只需要线路1和线路3从矿山返回终点时经过村庄即可。如果此时的解在此村庄有购买行为，就代表必须是线路2和线路4才可以通过，因为线路1和线路3并不会在此村庄补给；反之，如果解在此村庄的购买量为0，则线路1，3和2，4完全等价。我们计算得到线路2和线路4每一时间段的资源花费值，如表1.2所示：

表1.2　路线2、4各时间段资源花费值

线路	时间段1消耗	时间段2消耗	时间段3消耗	时间段4消耗
2	(98,98)	(243,211)	(36,40)	\
4	(98,98)	(245,211)	(115,123)	(42,38)

将这些数据代入之前建立好的整数线性规划模型中，得到每条路线最后保留的资金以及补给数量如表1.3所示：

表1.3　路线2、4补给量、剩余资金

线路	资金	起点补给	村庄1补给	村庄2补给	村庄3补给
2	10470	(178,333)	(163,0)	(36,16)	\
4	9800	(180,330)	(163,0)	(157,140)	(0,0)

可见，最高资金为10470元。同时我们可以看到路径4中的最后一个连接着终点的村庄并未补给，即未产生作用。利用在起点和村庄每次的补给值，即可填写表格Result.xlsx。我们得到玩家的最优策略即玩家在起点购买178箱水和333箱食物，1-8天前往村庄，并在村庄补给163箱水和0箱食物，9-10天前往矿山，11-19天在矿山挖矿，其中17，18天在矿山休息，20-21天前往村庄并补给36箱水和16箱食物，22-24天

前往终点，如图1.11所示：

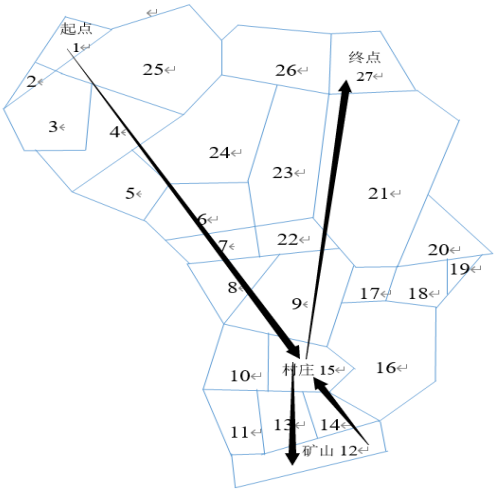


图1.11　第一关玩家最优策略示意图

5.1.2.2第二关分析与求解

关卡二的天气情况、各种基本变量都与关卡一一样，只是图发生了变化。因此，我们可以从关卡一的分析与结果中总结一些经验来帮助我们缩小考虑的范围。

首先，利用$floyd$算法，我们算出从起点到62号村庄的最小耗时为9天，到55号矿山的最小耗时为9天，到30号矿山的最小耗时为7天，到39号村庄的最小耗时为8天，且有地图所示会经过30号矿山。

如果我们仍以枚举k来计算的话，是十分复杂的，因为我们需要考虑两组不同的矿山与村庄。为此我们考虑从挖矿次数来分析。从关卡一我们可以得到，关卡一的最优可能性只会是一次挖矿或两次挖矿中产生。因此，我们在关卡二可以沿用此经验。当挖矿次数为1时，有两种情况即分别在30号挖和55号挖。当挖矿次数为2时，有4种情况：①两次均在30号挖；②先在30号挖，再去55号挖；③先在55号挖，再去 30号挖；④两次均在55号挖。他们对应的大致路线可以用图1.12表示。

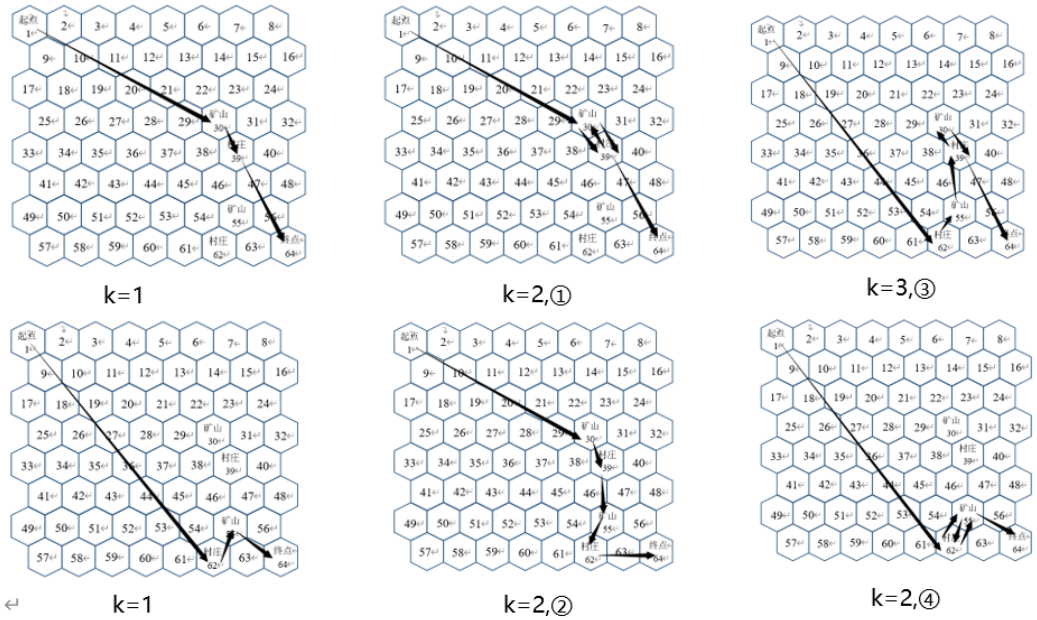


图1.12　关卡二各类路线图

显然，挖矿次数为2时，②先在30号挖，再去55号挖一定会优于③先在55号挖，再

去 30号挖，因为55号区域离终点更近。因此我们可以直接舍弃③这种情况。关卡2中，村庄和矿山的最近距离为1，较小。因此，挖矿次数为1时，可视作未充分利用挖矿赚钱，其最后的资金一定会比挖矿次数为2时的资金少，因此我们也可以舍弃挖矿次数为1的情况。

由第一关的第二和第四条路径可知，当玩家最后从矿山往终点走时，若有村庄在此最短路上，则玩家是不会在该村庄补给的。（有问题）

确定好这三种情况后，我们即确定了路线中矿山的数目以及位置号，于是我们根据k值，在可能出现村庄的时间段加入村庄，利用之前的贪心思想以及特定天气情况，我们得到了这些路线的局部最优时间分配，如图1.13所示：

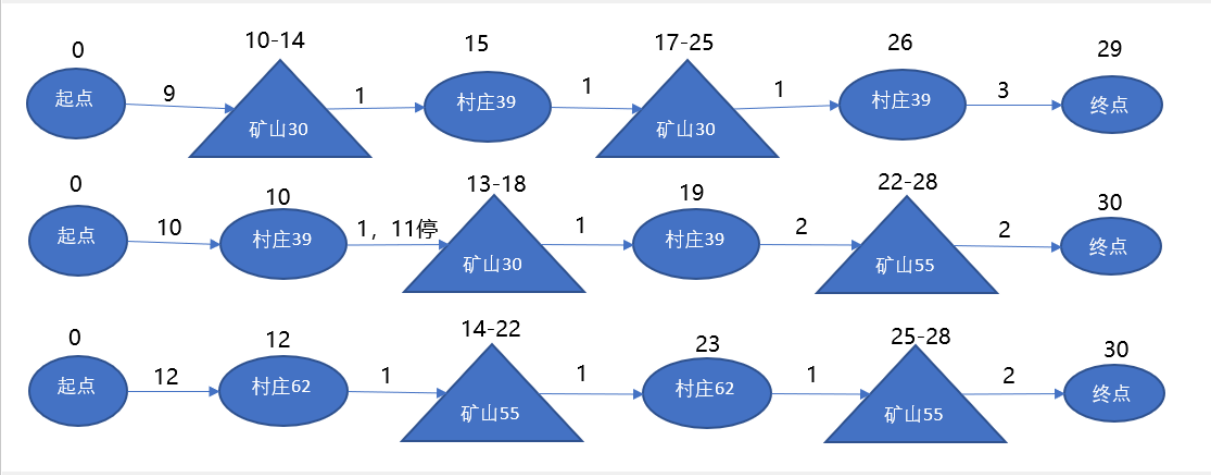


图1.13　三种路线局部最优时间分配图

同时，我们计算得到每一时间段的资源花费值，如表1.4所示：

表1.4　三种线路各时间段资源花费值

线路	时间段1消耗	时间段2消耗	时间段3消耗
1	(247, 227)	(239, 231)	(36, 40)
2	(130, 122)	(189, 169)	(196, 200)
3	(156, 144)	(236, 218)	(126, 128)

将这些数据代入之前建立好的整数线性规划模型中，得到每条路线最后保留的资金以及补给数量如表1.5所示：

表1.5　各线路补给量、剩余资金

线路	资金	起点补给	村庄1补给	村庄2补给
1	12345	(247, 229)	(246, 229)	(29, 40)
2	12730	(130, 405)	(211, 0)	(174, 86)
3	12460	(156, 366)	(252, 0)	(110, 124)

可见，最高资金为12730元。

利用在起点和村庄每次的补给值，即可填写表格Result.xlsx。我们得到玩家的最优策略如图1.14所示

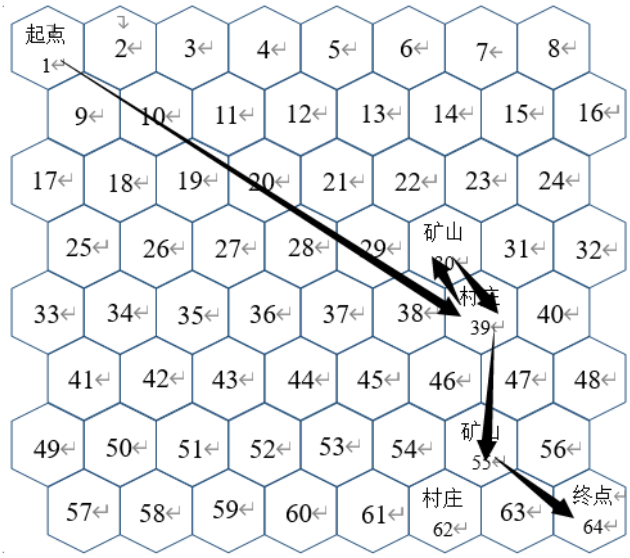


图1.14　第二关玩家最优策略示意图

即玩家即在起点购买130箱水和405箱食物，1-10天前往村庄39号，并在村庄补给211箱水和0箱食物，11-12天前往矿山30号，其中11天因沙暴停留，13-18天在矿山挖矿，19天前往村庄39号并补给174箱水和86箱食物，20-21天前往矿山55号，22-28天在矿山55号挖矿，29-30天前往终点。

5.2 问题二的模型及求解

5.2.1第三关模型建立及分析

（一）行走路线的决策

在第三关的地图下，在行走路线上玩家有两类选择方案，如图2.1所示：方案1为玩家选择不经过矿山直接到达终点，那么最短的路线为：1→4→6→13，记为路线1；方案2为玩家选择到矿山挖矿获取资金后再前往终点，那么其中一条最短的路线为1→4→3→9→11→13，记为路线2。

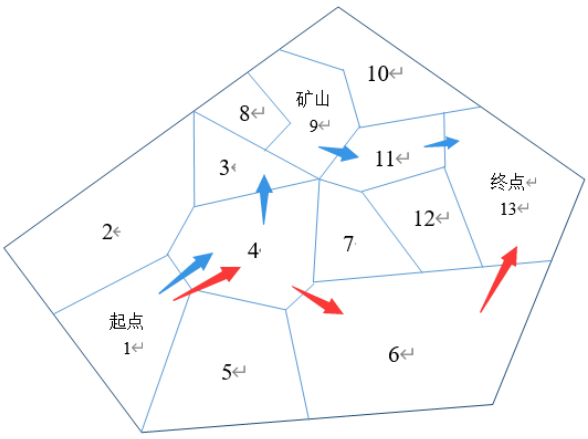


图2.1 方案1 、方案2路线

通过对地图的观察，我们发现如果玩家仅根据天气情况做出当天的行走方案，那么路线1中玩家从起点直接到终点与路线2中玩家从起点到矿山的天数一致，途中消耗

的水和食物的量也一致。为了达到玩家到达终点时所剩资金尽可能多这一目标，我们只需要讨论玩家在路线2中挖矿所得资金以及从矿山行走至终点期间所消耗的水、食物的价格的正负即可作出判断。

玩家在矿山挖矿获得资金的多少与天气状况有关，天气晴朗时玩家每天挖矿可以获得的净收益为200-165=35元，晴朗时挖矿天数越多所获资金越多；天气高温时玩家每天挖矿可以获得的净收益为200-405=-205元，即玩家在天气高温时挖矿会产生亏损，因此玩家可以通过当天的天气状况决定是否挖矿：

天气晴朗时在矿山的玩家会选择挖矿，天气高温时在矿山的玩家选择不挖矿。

为了判断玩家在路线2中挖矿所得资金以及从矿山行走至终点期间所消耗的水、食物的价格的正负，我们考虑一种极端情况：玩家在矿山挖矿5天，且这5天内天气晴朗，玩家从矿山行走至终点的时期内天气状况也为晴朗。该情况是玩家选择第二类方案可以获得最多收益的情况，因为相比高温，晴朗可以获得正的净收益，且行走时消耗的水、食物少，5天是玩家可以挖矿的最大天数，玩家挖矿5天可以获取最多的资金收益。在该情况下：

玩家在矿山挖矿5天获取的资金：200×5=1000元

消耗水、食物金额：（3×5+4×10）×3×5=825元

从矿山行走至终点消耗金额：（3×5+4×10）×2×2=220元

最终的净收益：1000-825-220=-45元

可以发现，在选择第二类方案能够取得最大净收益的情况下，其净收益相较于直接前往终点仍会产生损失，因此最佳的行走路线一定是1→5→6→13或1→4→6→13，如图2.2所示：

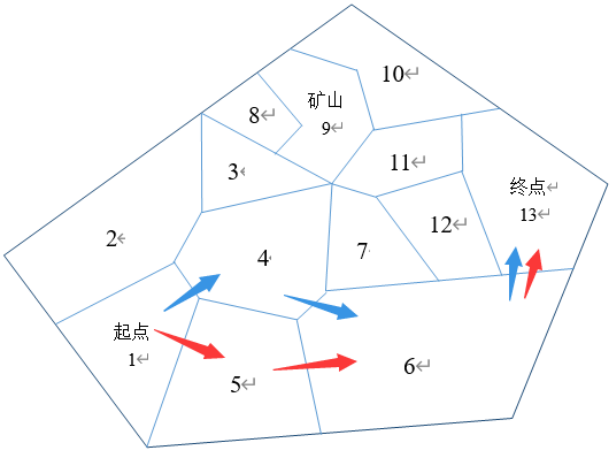


图2.2　最佳行走路线

（二）模型建立

由于玩家仅知道当天的天气状况，并据此决定当天的行动方案。假设改玩家记忆力很好，可以记得从第0天开始至当天的天气状况，那么我们就可以依照玩家经历过的这几天的天气来预测明天的天气情况，并做出相应的决策。但我们发现关卡三的截至时间是10天，而只需要跨过三片区域就可以到达终点。因此玩家已知道的数据量并不会太大。故我们考虑将天气的变化视作一个马尔可夫过程[2]，即明天的天气的情况只与今天有关，用条件概率表述如式（2.1）

$$P\{X(t)=x|X(t_n)=x_n,\ldots,X(t_1)=x_1\}=P\{X(t)=x|X(t_n)=x_n\} \quad (2.1)$$

其中$X(t)$为离散型随机变量。

针对第三关，当今天的天气为晴朗时，玩家必须选择直接跨越区域；而当今天的天气为高温时，玩家则有两种选择，一种是在原区域等待，另一种是高温穿越区域。记晴朗天气为A，高温天气为B，有了马尔可夫过程处理后，我们只需要利用一个样本数据来得到条件概率$P(A|B)$和$P(B|B)$，将概率值更大的那个视作明天的天气情况，来帮助玩家做出决策。为此，我们利用问题一中的天气情况作为样本数据，来得到这两个概率，进而分析玩家的决策。

5.2.2第三关模型求解及结果

玩家若每天都行走则最快第3日到达终点，我们发现当第i天天气为高温、玩家处于A区域时，为到达与之相邻的B区域，玩家可以有两种决策：（1）行走，（2）停留。

（1）当玩家选择行走时，玩家从A区域行走至B区域，花费资源的价格为270元

（2）当玩家选择停留时，第i+1天的天气有两种情况：晴朗和高温。如果第i+1天天气为晴朗，则玩家行走至B区域花费的资源的价格为245元；如果第i+1天天气为高温，则玩家行走至B区域花费的资源的价格为405元；记X_i=A、B、C表示第i天的天气状况分别为晴朗、高温、沙暴，根据附件1第1关的天气状况分布可以得到如下条件概率：

$$P\{X_{i+1}=A|X_i=B\}=5/14$$

$$P\{X_{i+1}=B|X_i=B\}=6/14$$

$$P\{X_{i+1}=C|X_i=B\}=3/14$$

记p(A)、p(B)、p(C)分别为晴朗、高温、沙暴天气状况下行走花费的资源价格，根据以上条件概率可以求得玩家在A区域选择先停留后行走时花费资源的期望价格E（P）为：

E（p）=p（A）$P\{X_{i+1}=A|X_i=B\}$+p(B)$P\{X_{i+1}=B|X_i=B\}$=332.272元。

通过比较玩家两种决策的收益我们可以发现：高温天气选择行走而不是停留将获得更高的收益，因此玩家不论遇到晴天还是高温都会在已有路线上选择行走。

综上，在第三关的情况下，玩家的最佳策略为：行走路线为1→5→6→13或1→4→6→13且玩家在任意一区域不停留。

5.2.2.1第四关数学模型建立

（一）行走路线的决策

表2.1　天气状况分布

日期	1	2	3	4	5	6	7	8	9	10
天气	高温	高温	晴朗	沙暴	晴朗	高温	沙暴	晴朗	高温	高温
日期	11	12	13	14	15	16	17	18	19	20
天气	沙暴	高温	晴朗	高温	高温	高温	沙暴	沙暴	高温	高温
日期	21	22	23	24	25	26	27	28	29	30
天气	晴朗	晴朗	高温	晴朗	沙暴	高温	晴朗	晴朗	高温	高温

我们利用第一关当中30天的天气情况,如上表所示:可以得出高温出现的次数与晴朗出现的次数的比例是15:9记为ρ。由于沙暴天气属于少数情况,所以这里暂不考虑沙暴的影响,对于第四关,首先我们对其做一个路径最优的策略,为了方便讨论,我们把晴天与高温的区别归约为一种天气状态P,在P状态下其消耗水和食物为高温与晴朗的值按比例ρ组成，具体见表2.2，同时我们大致P状态时行走和挖矿状态下消耗的资源重量，具体见表2.3。

表2.2　P状态下资源基础消耗量（箱）

物资品种	水	食物
一天消耗量(箱)	6.75	7.125

表2.3　P状态下资源消耗重量

行为方式	行走	挖矿
一天需要的重量(kg)	69	103.5

下面根据贪心思想给出4中不同路径方案：

一.　若从起点→矿山→终点:

假设玩家在起点处买尽量多的资源

(1)从起点出发到达村庄或者矿山会耗费5天,重量花费为:$5 \times 69 = 345\text{kg}$

(2)从矿山到终点需要3天,重量花费为:$3 \times 69 = 207\text{kg}$

(3)可在矿山工作的天数:

剩余资源重量:$1200 - 345 - 207 = 648\text{kg}$

因此可以在矿山工作:$648 \div 103.5 \approx 6$天

因此,需要水和食物分别为229.5箱、242.25箱

最终收益为:$16000 - 229.5 \times 5 - 242.25 \times 10 = 12430$元

二:起点→矿山(6天)→村庄→矿山(8天)→村庄→回终点

此问题可以转化为如问题一样的的整数线性规划问题，见式（1.13）

利用附录(五)代码解得各阶段补给资源数量、总收益，具体见下表：

表2.4　方案二资源补给量及总收益

	起点	一次村庄补给	第二次村庄补给	总收益
水(箱)	216	248	8.5	13140.0
食物(箱)	276	180	42.75	

三.若从起点→村庄→矿山→终点:

(1)从起点出发到5天，所花费资源重量为:$5 \times 69 = 345kg$

(2)村庄到矿山需要2天，所花费资源重量为:$2 \times 69 = 138kg$

(3)从矿山到终点需要3天，所花费资源重量为:$3 \times 69 = 207kg$

(4)可以在矿山挖矿天数为:$(1200 - 138 - 207) \div 105.5 \approx 8$天

同上代码可以解得各阶段补给资源数量、总收益，具体见下表：

表2.5　方案三各阶段资源补给量及总收益

	起点	一次村庄补给	总收益
水(箱)	190.3	106.6	12836.66
食物(箱)	314.5	0	

四:起点→村庄→矿山→村庄→矿山→终点

同上代码解得各阶段补给资源数量、总收益，具体见下表：

表2.6　方案四各阶段资源补给量及总收益

	起点	一次村庄补给	第二次村庄补给	总收益
水(箱)	168.5	115	229.5	14355.0
食物(箱)	347.25	0	194.25	

以上四种方案的详细情况如表2.7：

表2.7　四种方案行走、挖矿天数对比

方案	行走天数	挖矿天数
方案一	8天	6天
方案二	14天	14天
方案三	10天	8天
方案四	14天	16天

考虑到少数沙暴的影响,由于在行走过程上遇到沙暴不得不停留,我们考虑设置一个惩罚因子f,规定为在沙暴中停留一天消耗金钱200元；200元为全部在起点购买时的150元和全部在村庄购买时300元的中间值。

我们把第一问当中沙暴出现的频率近似为概率为20%，最终利用惩罚因子计算出每种方案的惩罚后收益，即最终收益，具体结果见表2.8：

表2.8　四种方案的惩罚后收益

	行走中期望出现沙暴天数	惩罚因子	惩罚代价	惩罚前收益	惩罚后收益
方案一	1天	200	200	12430	12230
方案二	3.5天	200	700	13140	12430
方案三	2.5天	200	500	12836.7	12336.7
方案四	3.5天	200	700	14355	13655

通过比较惩罚后收益，我们得到综合期望收益最大的是方案四:起点→村庄→矿山→村庄→矿山→终点。

（二）行走、停留、挖矿的决策

我们将天气情况分为两类:晴天和高温,将玩家所处区域分为两类：矿山和其他区域。

在“起点→村庄→矿山→村庄→矿山→终点”此路径规划中,若连续过多的挖矿,会导致水和食物过早消耗完而导致无法完成任务,因此有必要讨论在晴朗和高温情况下,矿区是否接着挖矿,停留,还是行走。

根据前面的分析，在此路线中从矿山行走到村庄需要2天时间,考虑极端情况：两天时间均为高温，这两天内消耗36箱水和36箱食物，由于停留消耗基础消耗量，挖矿消耗基础消耗量的3倍，因此我们可以得到高温和晴朗天气下各种行为的相对消耗量，具体见表2.9和表2.10。

表2.9　高温时各类行动资源相对消耗量

高温	挖矿	停留	走
相对消耗的水w	27	9	0
相对消耗食物f	27	9	0

表2.10 晴朗时各类行动资源相对消耗量

晴朗	挖矿	停留	走
相对消耗的水w	9	3	0
相对消耗食物f	12	4	0

加下来我们将根据天气情况讨论玩家选择行走、停留还是挖矿：

（一）当天是晴天:

（1） 在其他区域，玩家直接选择行走

（2） 在矿山：

① 在矿山 36<w<45 且 36<f<48，需要赶紧行走到村庄

②在矿山 w≥45 且 f≥48 就继续挖矿

（二）当天是高温

（1）在其他区域，根据第三关的期望讨论结果,玩家也不应该停留,直接行走

（2）在矿山

①若36<w<45 且 36<f<45，需要赶紧行走到村庄

②若45<w<73 且 36<f<73，此时需要讨论：

由第一问的天气分布得出条件概率:高温转高温为6/14,高温转晴朗为5/14:

此时停留的收益在于,若第二天是晴朗可以挖矿一天，分别计算停留的期望收益和行走的期望收益，得到：

（1）停留收益: (1000-9*10-12*20)*5/14-(9*30* 6/14)=123.5

（2）走期望收益:0

因此决策为停留一天

③若w>=73且f>=73，玩家继续挖矿

结论:（1）玩家的行走路线为: 起点→村庄→矿山→村庄→矿山→终点

（2）行走、停留还是挖矿的决策见下图2.2：

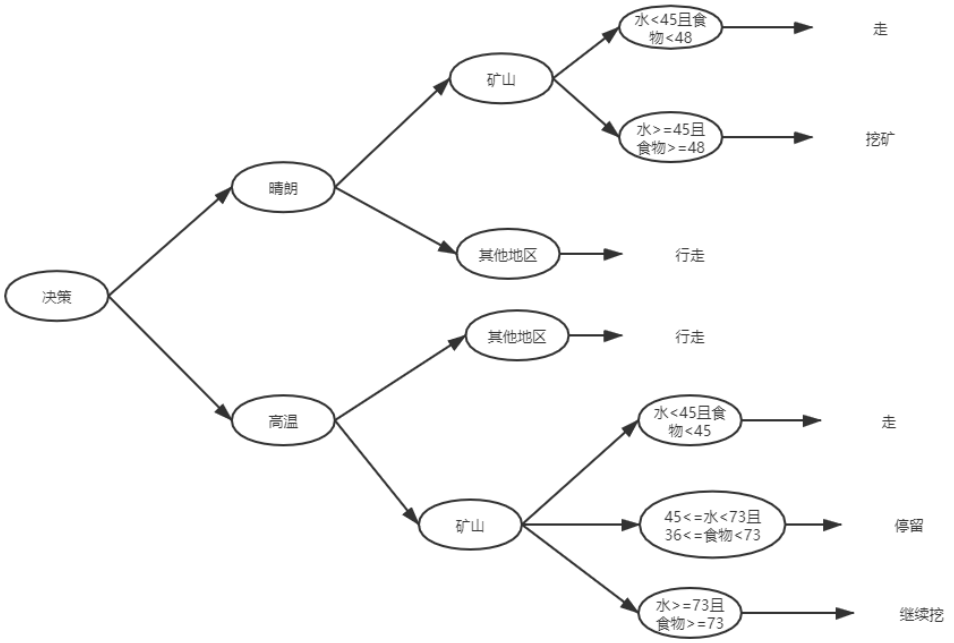


图2.2　玩家各类行为决策图

5.3问题三的模型及求解

5.3.1第五关

5.3.1.1第五关数学模型建立

由于两玩家如果在某天从A区域行走至B区域，行走、挖矿消耗的资源会大幅增加，挖矿获得的收益会大幅减少，因此玩家在制定行动方案时会量避免与其他玩家相撞以降低最终受益减少的风险，同时由于玩家可以事先知道所有的天气状况，选择在高温时等待、第二日晴朗时行走相比在高温时行走可以减少一部分资源的消耗，因此玩家可以根据天气状况调整自己的行走、停留抉择来避开与其他玩家相撞的可能性。

记某玩家在行走路线中总的资源消耗价格为M，我们定义“额外消耗”这一概念，记为M^*,记M'为某玩家为了降低与其他玩家相撞的风险采取新行动方案所消耗的资源价格，minM为该路线下消耗资源最小价格，额外消耗即为该玩家新行动方案消耗资源价格与该路线最小资源消耗价格的差额，表示为：

$$M^* = M' - minM \tag{3.1}$$

其中基于第三关中得出的结论，理性玩家不会前往矿山挖矿，而是选择直接前往终点，因此一条路线中总的资源消耗价格M包括行走消耗的资源价格和停留消耗的资源价格：

（1）基础消耗价格：

令晴朗时水的基础消耗量为c_{00}，食物的基础消耗量为C_{01}，;高温时水的基础消耗量为c_{10}，食物的基础消耗量为c_{11}，水的基准价格为p_0，食物的基准价格为p_1，则不同天气的基础消耗价格 C可以表示为：

$$C = \begin{cases} c_{00}p_0 + c_{01}p_1, 天气为晴朗 \\ c_{10}p_0 + c_{11}p_1, 天气为高温 \end{cases} \tag{3.2}$$

（2）总资源消耗价格：

引入0-1变量，该玩家第z天到达终点：

$$f_i = \begin{cases} 1, 第i天行走 \\ 0, 第i天停留 \end{cases}, i = 0,1 \dots, z - \tag{3.3}$$

$$h_i = \begin{cases} 1, 第i天停留 \\ 0, 第i天行走 \end{cases}, i = 0,1 \dots, z - 1 \tag{3.4}$$

假设n名玩家中，第i天有k（2≤k≤n）名玩家从A区域行走至B区域（A≠B），则某名玩家因行走消耗的资源价格为M_1，因停留消耗的资源价格为M_2，总的消耗资源价格M为：

$$\mathrm{M} = M_1 + M_2 = \sum_{i=0}^{z-1} f_i 2kC + \sum_{i=0}^{z-1} h_i C \tag{3.5}$$

为了减少同其他玩家在同一天前往同一区域造成的损失，玩家可以通过等停留等方式进行避让，但是因此带来的额外消耗要尽量小，故目标函数为：

$$\min M^* \tag{3.6}$$

最终建立如下规划方程：

$$\min M^*$$
$$.\mathrm{t.}\begin{cases} \mathrm{M} = M_1 + M_2 = \sum_{i=0}^{z-1} f_i 2kC + \sum_{i=0}^{z-1} h_i C, k \in [2, n] \\ M^* = M' - minM \\ C = \begin{cases} c_{00}p_0 + c_{01}p_1, \textit{天气为晴朗} \\ c_{10}p_0 + c_{11}p_1, \textit{天气为高温} \end{cases} \\ f_i = \begin{cases} 1, \textit{第}i\textit{天行走} \\ 0, \textit{第}i\textit{天停留} \end{cases}, i = 0,1 \ldots, z-1 \\ h_i = \begin{cases} 1, \textit{第}i\textit{天停留} \\ 0, \textit{第}i\textit{天行走} \end{cases}, i = 0,1 \ldots, z-1 \end{cases} \tag{3.7}$$

5.3.1.2求解模型

具体分析第五关，基于我们在第三关的结论：理性的玩家不会去矿山挖矿，而是选择直接前往终点，因此我们从终点回溯，选择了4条路径较短的路径，这是玩家最有概率选择的路径，分别为：1→4→6→13、1→5→6→13、1→4→7→11→13、1→4→7→12→13。

另作如下符号含义表：

表3.1　符号含义表

符号	含义
A	天气晴朗，玩家选择行走
B	天气高温，玩家选择行走
A	天气晴朗，玩家选择停留
B	天气高温，玩家选择停留

如：ABAA表示玩家在高温时停留，在晴朗时行走。

根据附件材料，在截止期限内的天气状况可以表示为：ABAAAABBBB，接下来我们分析每条线路的最小“额外消耗”：

（1）1→4→6→13与1→4→6→13或1→5→6→13与1→5→6→13：

考虑甲玩家与乙玩家都选择1→4→6→13这条道路，该线路的最优方案为：ABAA，

消耗资源金额为465元。甲玩家为了避免与乙玩家走完全相同的道路导致每天都遭受额外的损失可能会尽早做出避让，因此可能做出如下策略：第一天停留，第二天行走或停留，分别对应方案①ABAA和方案②ABAAA，该种情况共有三种方案：①ABAA、②ABAA、③ABAAA。经计算方案①消耗物资金额为465元，对应路线最少消耗金额为465元；方案②消耗物资金额为545元，对应路线最少消耗金额为465元；方案③消耗物资金额为520元，对应路线最少消耗金额为465元，详细结果见表3.2。

表3.2　额外消耗表1

方案序号	行动方案	额外消耗（元）
①	ABAA(1→4→6→13、乙)	0
②	ABAA（1→4→6→13、甲）	80
③	ABAAA（1→4→6→13、甲）	55

从上表我们可以看出，甲玩家与乙玩家行走路线均为1→4→6→13时甲玩家做出避让最小的额外消耗为55元，另一玩家额外消耗为0元。同理，甲玩家与乙玩家行走路线均为1→5→6→13时甲玩家做出避让最小额外消耗为55元，乙玩家为0元。

当然避让方也可以为乙，即乙避让最小额外消耗为55元，甲为0元。总而言之，双方路线均为1→4→6→13或1→5→6→13时，总是一方最小额外消耗量为0，一方最小额外消耗量为55元。

（2）1→4→6→13与1→5→6→13：

首先考虑甲玩家选择路线为1→4→6→13，该玩家需要避免与选择1→5→6→13的乙玩家路线相撞，1→5→6→13的最优行动方案为ABAA，消耗资源金额为465元。甲玩家为了不在后期与乙玩家相撞，需要将时间错开，由于前两天的路线没有重合，因此甲玩家可以选择在第二日高温时行走，保证后期不会与乙玩家碰撞，具体方案为ABA，消耗资源金额为490元，该种情况下共有两种方案：

①ABAA（1→5→6→13），消耗资源金额为465元，对应路线最少消耗资源金额为465元；②ABA（1→4→6→13），消耗资源金额为490元，对应最佳路线消耗资源金额为465元，详细结果见表3.3。

表3.3　额外消耗表2

方案序号	行动方案	额外消耗（元）
①	ABAA（1→5→6→13、乙）	0
②	ABA（1→4→6→13、甲）	25

根据上表显示，甲玩家行走路线为1→4→6→13、乙玩家行走路线为1→5→6→13时甲玩家做出避让最小的额外消耗为25元，而乙玩家额外消耗为0元；若两玩家交换线路，则甲玩家行走路线为1→5→6→13，额外消耗为0元，乙玩家额外消耗为25元。

也可以由乙方担任避让方，总而言之，若一方选择1→4→6→13并且按ABA方式行走，另一方选择1→5→6→13并且按ABAA方式行走，就可以做到前者最小额外消耗为0元，后者最小额外消耗为25元。

（3）1→4→6→13与1→4→7→11→13/1→4→7→12→13：

考虑甲玩家选择路线为1→4→6→13，其要避免与选择路线为1→4→7→11→13的乙玩家在第一天相撞，1→4→7→11→13的最优行动方案为ABAAAA，消耗资源金额为685元。甲玩家第一天就要避免与其他玩家碰撞，因此第一天需要在原地停留，第二天

可以选择停留或行走，对应方案分别为：方案①ABAA，方案②ABAAA，该种情况下共有三种方案：

①ABAAAA（1→4→7→11→13），消耗资源金额为685元，对应最佳路线消耗资源金额为685元；②ABAA，消耗资源金额为545元，对应最佳路线消耗资源金额为465元；③ABAAA，消耗资源金额为520元，对应最佳路线消耗资源金额为465元，详细结果见表3.4。

表3.4　额外消耗表3

方案序号	行动方案	额外消耗（元）
①	ABAAAA（1→4→7→11→13、乙）	0
②	ABAA（1→4→6→13、甲）	80
③	ABAAA（1→4→6→13、甲）	55

根据上表显示，甲玩家行走路线为1→4→6→13、另一玩家行走路线为1→4→7→11→13时甲玩家做出避让最小额外消耗为55元，乙玩家额外消耗为0元。但是由于1→4→7→11→13线路过长，其最小消耗金额为685元，相比情形（1）（2）难以保证到达终点所剩资金余额尽量多的目标，因此可以不考虑1→4→7→11→13、1→4→7→12→13两种情况。

5.3.1.3结果

将（1）（2）两种情形的额外消耗金额与资金余额进行对比，形成表3.5：

表3.5　（1）（2）情形对应的额外消耗金额与消耗金额

情形	额外消耗金额（元）	消耗金额（元）
（1）	55（甲）、0（乙）	520（甲）、465（乙）
（2）	25（甲）、0（乙）	490（甲）、465（乙）

通过比较（1）、（2）可以发现情形（2）的方案可以使玩家途中消耗金额最小、额外消耗金额最小，因此情形（2）所列方案为最佳策略。综上所述，在第五关、玩家数为2且可以预知全部天气的情况下，玩家的最佳策略为一人选择1→4→6→13路线，另一人选择1→5→6→13路线，并且其中一方采取ABA行动方案（第一天、第二天、第三天均行走）、另一方采取ABAA行动方式（第一天行走、第二天停留、第三天、第四天行走）。

5.3.2第六关

5.3.2.1模型建立

首先，我们假设第六关中的三名玩家代号 1、2、3 均想要走最优路线，即第四关中我们求得的起点→村庄→矿山→村庄→矿山→终点，这一最优路线有很多走法，由于玩家在某天从 A 区域行走至 B 区域（A≠B）（记为情况 1）或者同时在矿山挖矿（记为情况 2）都会给自己带来额外的损失，因此玩家需要根据行动结束后其他玩家当天的行动策略、剩余资源数量以及第二日的天气状况来确定自己第二日的行程，这样做的目的是与其他玩家避开同时从 A 区域前往 B 区域或者同时在矿山挖矿带来的额外风险损失，因此玩家确定行动方案的原则包括：（1）减少额外损失；（2）使剩余收益尽可能大。

我们定义不同天气状况下基础消耗量价格 C 的表达式：

$$C=\begin{cases}c_{00}p_0+c_{01}p_1, 天气为晴朗\\ c_{10}p_0+c_{11}p_1, 天气为高温\\ c_{20}p_0+c_{21}p_1, 天气为沙暴\end{cases} \quad (3.8)$$

我们分以下三种情况讨论：

（1）玩家经分析发现下一天不会与其他玩家冲突（情况 1、情况 2 都不会发生）：

此时该玩家不需要在路线上做出避让，只需要根据天气来判断下一天的行动安排，具体安排可参见第四关结果；

（2）玩家经分析发现下一天有概率p_1会与 k 名玩家发生情况 1：

此时该玩家有两种方案，方案一：该玩家在原地停留一天后继续按原路线前行；方案二：该玩家不考虑相撞风险按计划行事。

该玩家选择方案一的收益函数为：$V_{11=}-p_1C+(1-p_1)C$

该玩家选择方案二的收益函数为：$V_{12}=-p_1 2kC-\left(1-p_1\right)2C$

同理，代号 2、3 的玩家也可以按此计算收益函数V_{21}、V_{22}、V_{31}、V_{32}

以此来构建三人博弈表：

表3.1：三人博弈表一

	玩家2方案一		玩家2方案二	
	玩家3方案一	玩家3方案二	玩家3方案一	玩家3方案二
玩家1方案一	（V_{11},V_{21},V_{31}）	（V_{11},V_{21},V_{32}）	（V_{11},V_{22},V_{31}）	（V_{11},V_{22},V_{32}）
玩家1方案二	（V_{12},V_{21},V_{31}）	（V_{12},V_{21},V_{32}）	（V_{12},V_{22},V_{31}）	（V_{12},V_{22},V_{32}）

通过解该表可以得到情况1下玩家的最优选择策略。

（3）玩家经分析发现下一天有p_2的概率会与k名玩家发生情况2：

以该玩家此时正在矿山挖矿为例，假若本玩家下一天本应继续挖矿，如果分析将会有k名玩家来到矿山挖矿，则该玩家有两种选择方案，方案一：该玩家停止挖矿；方案二：该玩家在原地继续挖矿。

该玩家选择方案一的收益函数为：$V_{11}^{'}=-p_2C+(1-p_2)1000$

该玩家选择方案二的收益函数为：$V_{12}^{'}=p_2\left(\frac{C}{k}-3C\right)+(1-p_2)(1000-3C)$

同理，代号2、3的玩家也可以按此计算收益函数$V_{21}^{'}$、$V_{22}^{'}$、$V_{31}^{'}$、$V_{32}^{'}$

以此构建三人博弈表：

表3.2：三人博弈表二

	玩家2方案一		玩家2方案二	
	玩家3方案一	玩家3方案二	玩家3方案一	玩家3方案二
玩家1方案一	（$V_{11}^{'},V_{21}^{'},V_{31}^{'}$）	（$V_{11}^{'},V_{21}^{'},V_{32}^{'}$）	（$V_{11}^{'},V_{22}^{'},V_{31}^{'}$）	（$V_{11}^{'},V_{22}^{'},V_{32}^{'}$）
玩家1方案二	（$V_{12}^{'},V_{21}^{'},V_{31}^{'}$）	（$V_{12}^{'},V_{21}^{'},V_{32}^{'}$）	（$V_{12}^{'},V_{22}^{'},V_{31}^{'}$）	（$V_{12}^{'},V_{22}^{'},V_{32}^{'}$）

通过解该表可以得出情况2下玩家的最佳决策。

5.3.2.2模型求解及结果

在第六关中，对于玩家判断不会与其他玩家发生冲突的情况不再考虑，可以参加第四关的回答，第六关重点关注玩家判断会与其他玩家发生冲突的情况：

（1）发生情况1：
此时以天气晴朗为例，计算得出：

$$V_{11} = V_{21} = V_{31} = 55 - 110p_1$$
$$V_{12} = V_{22} = V_{32} = 110 - 220p_1$$

显然选择方案一的收益函数大于选择方案二的收益函数，通过划线法解三人博弈表，得到最优的解为玩家选择方案一。由于天气高温、沙暴时基础消耗量更多，因此这两种天气情况下玩家也应选择在原地停留一天。
（2）发生情况2：
此时以天气晴朗、k=2为例，计算得出：

$$V_{11}^{'} = V_{21}^{'} = V_{31}^{'} = 1000 - 1055p_2$$
$$V_{12}^{'} = V_{22}^{'} = V_{32}^{'} = 697.5p_2 - 835$$

因为$p_2 > 0$，所以可证$V_{11}^{'} > V_{12}^{'}$，即选择方案一的收益函数大于选择方案二的收益函数，通过划线法解三人博弈表，得到最优的解为玩家选择方案一。由于天气高温、沙暴时玩家挖矿的成本更高，净收益更少；k=3时3人同挖矿山带来的额外损失更多，因此各种天气下以及k=2或3的情况下，玩家预测会有新的玩家来挖矿山时要停止挖矿山，让给新玩家挖。

六、模型评价

6.1模型优点

(1)在问题一当中利用贪心思想和整数线性规划配合,解出每一种路径情况下的最优物资补给分配,通过贪心方案找到几种路径方案中比较出最优的的方案。
(2)在问题二当中,第三关通过极端测试比较体验证出最佳路径,第四关利用对天气进行规定统一,并按照问题一思路给出最优路径,然后再根据当前天气情况以及自身物资数量做具体决策分析。

6.2模型缺点

(1)在问题一的模型当中无法枚举出所用方案进行比较,使得讨论不够全面。
(2)在问题二当中所给出的决策都是固定的,在实际应用当中可能不够奏效。

6.3模型改进方向

(1)对于问题一进一步优化算法,能够全局搜索最优解,或者数学归纳证明出贪心策略全局正确性
(2)针对问题二将全局决策进一步优化为动态决策过程,使得模型更加全面

参考文献

[1]刘汝佳<<算法竞赛入门经典>>第二版　北京,清华大学出版社　2014年

[2]司守奎<<数学建模算法与应用>>第2版　北京　国防工业出版社　2015年

附录

附录一:最短路矩阵代码:

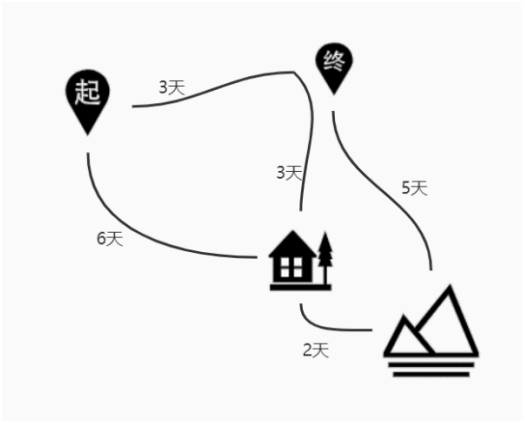
```
import networkx as nx
import numpy as np
# 此代码利用floyd算法求解出最短距离矩阵D
def func1(node,edge):
    g = nx.Graph()
    g.add_nodes_from(node)
    g.add_edges_from(edge)
    nx.draw(g,with_labels = True)
    A = np.array(nx.adjacency_matrix(g).todense())
    d = nx.floyd_warshall_numpy(g)
    return A,d
edge1 = [(1,2),(1,25),
         (2,3),
         (3,4),(3,25),
         (4,5),(4,25),(4,24),
         (5,6),(5,24),
         (6,7),(6,23),(6,24),
         (7,8),(7,22),
         (8,9),(8,22),
         (9,10),(9,15),(9,16),(9,17),(9,21),
         (10,11),(10,13),(10,15),
         (11,12),(11,13),
         (12,13),(12,14),
         (13,14),(13,15),
         (14,15),(14,16),
         (15,16),
         (16,17),(16,18),
         (17,18),(17,21),
         (18,19),(18,20),
         (19,20),
         (20,21),
         (21,22),(21,23),(21,27),
```

```
        (22,23),
        (23,24),(23,26),
        (24,25),(24,26),
        (25,26),
        (26,27)
       ]
node1 = np.array(range(1,27))
a1,d1 = func1(node1,edge1)
def legal(i,j,id):
    if(id>64 or id<1):
        return False
    if(j==1):
        if(id>=(i-1)*8+2 and id<=i*8):
            return True
        else:
            return False
    if(j==2):
        if(i%2==0):
            if(id>=8*i+1 and id<=8*i+1+6):
                return True
            else:
                return False
        else:
            if(id>=8*i+1 and id<=8*i+1+6):
                return True
            else:
                return False
    if(j==3):
        if(i%2==0):
            if(id>=8*i+2 and id<=8*i+2+6):
                return True
            else:
                return False
        else:
```

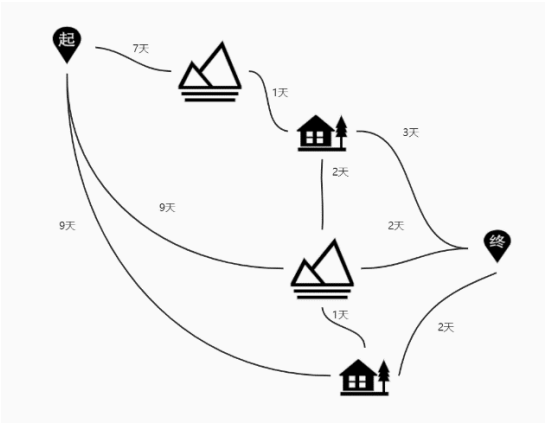
```
            if(id>=8*i+1 and id<=8*i+7):
                return True
            else:
                return False

def bound(x):
    if(x>=1 and x<=8):
        return 1
    if(x>=9 and x<=16):
        return 2
    if(x>=17 and x<=24):
        return 3
    if(x>=25 and x<=32):
        return 4
    if(x>=33 and x<=40):
        return 5
    if(x>=41 and x<=48):
        return 6
    if(x>=49 and x<=56):
        return 7
    if(x>=57 and x<=64):
        return 8

edge2 = []
for i in range(1,65):
    x = bound(i)
    if(x%2!=0):
        if(legal(x,1,i+1)):
            edge2.append((i,i+1))
        if(legal(x,2,i+7)):
            edge2.append((i,i+7))
        if(legal(x,3,i+8)):
            edge2.append((i,i+8))
    else:
```

```
            if(legal(x,1,i+1)):
                edge2.append((i,i+1))
            if(legal(x,2,i+8)):
                edge2.append((i,i+8))
            if(legal(x,3,i+9)):
                edge2.append((i,i+9))

 node2 = np.array(range(1,65))
a2,d2 = func1(node2,edge2)
```

附录二:第一、二、三、四关的对偶图:

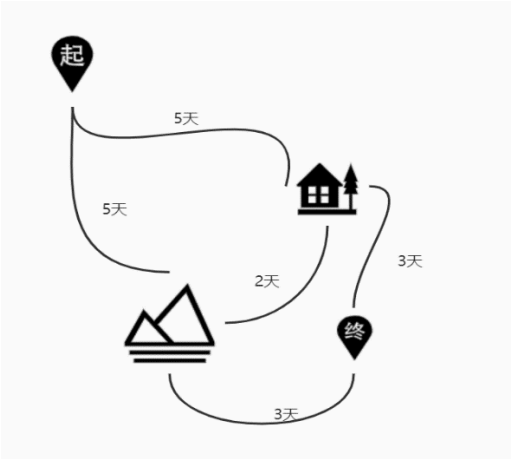


第一关



第二关

第三关



第四关

附录三:计算路程、挖矿消耗代码

```
 #!/usr/bin/env python
 # coding: utf-8
```

#此代码用于计算fwalk和fmine 因为测试数据过多，未给出测试数据，可以利用函数做测试。得到走路或挖矿的消耗。

```
weather = [
    2,2,1,0,1,2,0,1,2,2,
    0,2,1,2,2,2,0,0,2,2,
    1,1,2,1,0,2,1,1,2,2
]

w1 = 5
w2 = 8
f1 = 7
f2 = 6
ws = 10
fs = 10

kg1 = 29
kg2 = 36
kg3 = 50

def fwalk(t0,t1,weather):
    count = t0
    kgc = 0
    wc = 0
    fc = 0
    while(count !=t1):
        if(weather[count]==1):
            kgc = kgc + kg1*2
            wc = wc +2*w1
            fc = fc + 2*f1
        elif(weather[count]==2):
            kgc = kgc + kg2*2
            wc = wc +2*w2
            fc = fc + 2*f2
```

```
        else:
            kgc = kgc + kg3
            t1 = t1 + 1
            wc = wc +ws
            fc = fc + fs
        count = count + 1
    return t1,kgc,wc,fc


def mine_time(t2,kg,weather):
    while(kg>=0):
        if(weather[t2-1]==1):
            kg = kg - 3*kg1
        elif(weather[t2-1]==2):
            kg = kg - 3*kg2
        else:
            kg = kg - 3*kg3
        print(kg)
        t2 = t2 + 1
    return t2-1


def fmine(t0,t1,weather):
    count = t0
    kgc = 0
    wc = 0
    fc = 0
    while(count !=t1):
        if(weather[count]==1):
            kgc = kgc + kg1*3
            wc = wc +3*w1
            fc = fc + 3*f1
        elif(weather[count]==2):
            kgc = kgc + kg2*3
```

```
            wc = wc +3*w2
            fc = fc + 3*f2
        else:
            kgc = kgc + 3*kg3
            #t1 = t1 + 1
            wc = wc +3*ws
            fc = fc + 3*fs
        count = count + 1
    return t1,kgc,wc,fc
```

附录四:路径方案的检验代码:

```
#include<iostream>
using namespace std;
int type[31]={0,1,1,0,2,0,1,2,0,1,1,2,1,0,1,1,1,2,2,1,1,0,0,1,0,2,1,0,0,1,1};
int war[3]={5,8,10};
int food[3]={7,6,10};
int main(){
      int n=8;
      int sum1=0,sum2=0;
      int start;
      int end;
      for(int i=start;i<=end;i++){
       sum1+=war[type[i]];
       sum2+=food[type[i]];
      }
      sum1*=3;
      sum2*=3;
      cout<<sum1<<" "<<sum2<<endl;//从第start天挖矿到第end天水和食物所需水和食物箱数
      cout<<sum1*3+sum2*2<<endl; // 第start天挖矿到第end天水和食物所需水和食物总重量
      cout<<endl;   cout<<endl;   cout<<endl;

       sum1=0;sum2=0;
       int start1;
       int end1 ;
```

```
        for(int i=start1;i<=end1;i++){
               if(type[i]==2) {
        sum1+=war[type[i]];
        sum2+=food[type[i]];
        }
        else{
        sum1+=war[type[i]]*2;
        sum2+=food[type[i]]*2;
        }
       }
        cout<<sum1<<" "<<sum2<<endl;//从第start1天行走到第end1天水和食物所需水和食物箱数
        cout<<sum1*3+sum2*2<<endl;// 第start1天行走到第end1天水和食物所需水和食物总重量
        cout<<endl;   cout<<endl;   cout<<endl;

        return 0;
}
```

附录五:已知路径,线性规划解补给分配代码

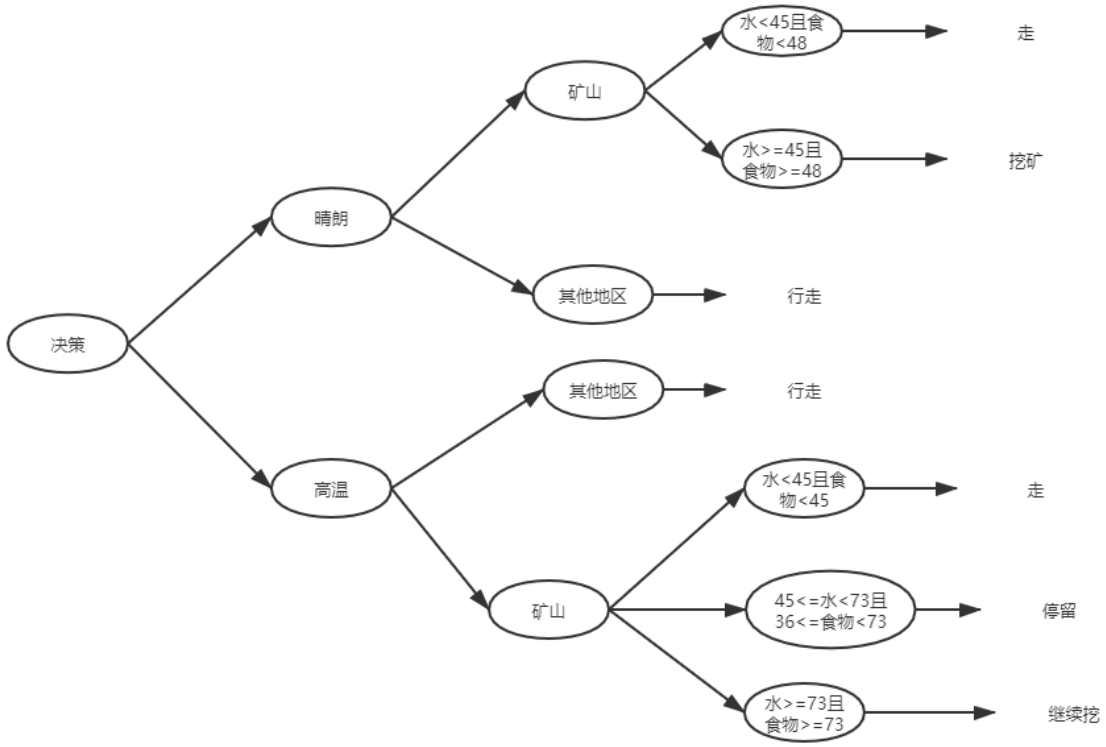
```
 #!/usr/bin/env python
 # coding: utf-8

 # k=2的四组结果 使用时，将某一组c取消注释，并更改m的值，再运行即可求解线性规划模型。

 import cvxpy as cp
 import numpy as np
 c=[
     #case2: m=[9000,4000] 12460
     #[156,10+210+16,10+84+32],
     #[144,14+192+12,14+90+24]

     #case2:m=[5000,9000] 12345
     #[114+117+16,16+207+16,36],
     #[110+105+12,12+207+12,40]

     #case2 m=[6000,7000] 12730
```

```
    #[130,10+16+147+16,26+138+32],
    #[122,10+12+135+12,26+150+24]

    #case1 m=[7000,0] 10470
    #[98,243,36],
    #[98,211,40]
]
c = np.array(c)
m=[[9000,4000]]
m=np.array(m)

x = cp.Variable((2,3),integer=True)
obj = cp.Maximize(10000-5*x[0,0]-10*x[1,0]-10*(x[0,1]+x[0,2])-
      20*(x[1,1]+x[1,2])+m[0,0]+m[0,1]
                  +2.5*(x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2]-c[0,2])
                  +5*(x[1,0]-c[1,0]+x[1,1]-c[1,1]+x[1,2]-c[1,2]))
cons = [
    3*x[0,0]+2*x[1,0]<=1200,
    5*x[0,0]+10*x[1,0]<=10000,
    x[0,0]-c[0,0]>=0,
    x[0,0]-c[0,0]+x[0,1]-c[0,1]>=0,
    x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2]-c[0,2]>=0,
    x[1,0]-c[1,0]>=0,
    x[1,0]-c[1,0]+x[1,1]-c[1,1]>=0,
    x[1,0]-c[1,0]+x[1,1]-c[1,1]+x[1,2]-c[1,2]>=0,
    3*(x[0,0]-c[0,0]+x[0,1])+2*(x[1,0]-c[1,0]+x[1,1])<=1200,
    3*(x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2])+2*(x[1,0]-c[1,0]+x[1,1]-c[1,1]+x[1,2])<=1200,
    x>=0
]
prob = cp.Problem(obj,cons)
prob.solve(solver = 'GLPK_MI')

print(x.value)
print(prob.value)
```

```
#!/usr/bin/env python
# coding: utf-8

# 第一关的k=3的测试结果，直接运行即可。

import cvxpy as cp
import numpy as np

c=[
      #case1: m=10000 9800
    [98,245,115,42],
    [98,211,123,38]
]
c = np.array(c)



x = cp.Variable((2,4),integer=True)
obj = cp.Maximize(-5*x[0,0]-10*x[1,0]-10*(x[0,1]+x[0,2]+x[0,3])-
      20*(x[1,1]+x[1,2]+x[1,3])+20000
                  +2.5*(x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2]-c[0,2]+x[0,3]-c[0,3])
                  +5*(x[1,0]-c[1,0]+x[1,1]-c[1,1]+x[1,2]-c[1,2]+x[1,3]-c[1,3]))
cons = [
    3*x[0,0]+2*x[1,0]<=1200,
    5*x[0,0]+10*x[1,0]<=10000,
    x[0,0]-c[0,0]>=0,
    x[0,0]-c[0,0]+x[0,1]-c[0,1]>=0,
    x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2]-c[0,2]>=0,
    x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2]-c[0,2]+x[0,3]-c[0,3]>=0,
    x[1,0]-c[1,0]>=0,
    x[1,0]-c[1,0]+x[1,1]-c[1,1]>=0,
    x[1,0]-c[1,0]+x[1,1]-c[1,1]+x[1,2]-c[1,2]>=0,
    x[1,0]-c[1,0]+x[1,1]-c[1,1]+x[1,2]-c[1,2]+x[1,3]-c[1,3]>=0,

    3*(x[0,0]-c[0,0]+x[0,1])+2*(x[1,0]-c[1,0]+x[1,1])<=1200,
    3*(x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2])+2*(x[1,0]-c[1,0]+x[1,1]-c[1,1]+x[1,2])<=1200,
```

```
        3*(x[0,0]-c[0,0]+x[0,1]-c[0,1]+x[0,2]-c[0,2]+x[0,3])+2*(x[1,0]-c[1,0]+x[1,1]-
          c[1,1]+x[1,2]-c[1,2]+x[1,3])<=1200,

        x>=0


    ]

    prob = cp.Problem(obj,cons)

    prob.solve(solver = 'GLPK_MI')


    print(x.value)

print(prob.value)
```

附录六:第四关决策图:



第四关决策图

附录七:支撑材料的文件目录:

(1) 全部源程序代码

(2) 第四关决策图

(3)一到四关对偶图

(4)结果文件result.xlsx