| 所属类别 | 2022 年“华数杯”全国大学生数学建模竞赛 | 参赛编号 |
| --- | --- | --- |
| 本科生组 | | CM2202870 |

# 基于整数线性规划及双链量子遗传算法对 WPCR 生产规划问题的研究


## 摘要

自来水管道清理机器人使用推广创造了巨大的需求缺口,本文运用了**整数线性规划**、**量子遗传算法**和**马尔科夫分析**等方法建立数学模型,分析给出了机器人工厂的最优生产方案,为工厂迎合市场需求提供有益指导。

针对问题一:对于具有明显线性关系的 WPCR 机器人组件组装工艺,我们以“使总成本最低”为目标函数,创新性的设计了**“产品产量矩阵”**,建立**整数线性规划**模型。该模型构建了**总成本函数**线性方程组,并对产品产量矩阵给出 56 条不等式约束,最终规划出了一种最优生产方案: 周日应组装 WPCR 机器人 28 台、A 组件 83 件、B 组件 110 件、C 使用组件 140 件,生产准备费用 385140 元,库存费用 0 元等,结果详见表 3。

针对问题二:本题在问题一的基础上新增了工艺流程约束,但总成本函数人具有强线性关系。我们沿用“总成本最低”目标函数和“产品产量矩阵”,对整数线性规划模型进行改进。通过**调整总成本函数方程组**和 56 条不等式约束,重新规划适应新条件的最优生产方案: 周一应组装 WPCR 机器人 34 台、A 组件 117 件、B 组件 34 件、C 使用组件 195 件,生产准备费用 519040 元,库存费用 25369.5 元等,结果详见表 4。

针对问题三:基于前述研究,WPCR 机器人的工时限制因新增检修维护而放宽,其组件生产总成本不再是纯线性关系,但仍以“总成本最低”为目标函数，可保留了“产品产量矩阵”。对于非线性优化问题,我们基于**双链量子遗传算法**,结合生产总成本改进函数,建立生产方案智能优化模型,演化出较优秀的生产检修方案:**总成本 113819713.3 元**、**第一次检修在**第 1 周周二(**第 2 天**)、**第二次检修在**第 3 周周四(**第 18 天**)...详见表 5。

针对问题四：承接问题二的研究基础,为更贴近订单情报实际情况,我们对历史的周订单数据进行了**马尔科夫分析**,得出表 8 所示的马尔科夫状态概率表。根据各个状态在周一到周日发生的不同概率,通过**计算机模拟**生成了 100000 组周订单需求,筛选出最符合要求且总成本最低的需求情况,最终得出最妥当的生产方案,使得**每天**的 WPCR 订单**正常交付率高于 95%，周正常交付率高于 85%**:周一应组装 WPCR 机器人 43 台、A 组件 129 件、B 组件 172 件、C 使用组件 215 件,生产准备费用 560020 元,库存费用 26081.5 元等,结果详见表 10。



**关键字:**组件生产 多变量线性规划 量子遗传算法 马尔科夫分析 计算机模拟

# 一、 问题重述

## 1.1 问题背景

自来水管道清洁机器人 (简称 WPCR) 因其保障管道通畅的高可靠性受到市场青睐,某工厂为争夺市场地位计划优化其 WPCR 机器人的生产计划。生产一台 WPCR 机器人的零部件涵盖: 容器艇 (A)、机械臂 (B)、动力系统 (C) 共 3 大类 8 种类型——共计 182 个零部件。该工厂每次生产计划的计划期为一周（即每次按照每周 7 天的订购数量实行订单生产），最终只有产品 WPCR 有出售, 容器艇、机器臂、动力系统不外销。为了提高关键设备利用率, 需严格控制总生产工时,A,B,C 类大组件都有工时限制。生产一台 WPCR 机器人所需的组件清单及其组装工时要求见下表 4 所示:

表 **1** 生产一台 **WPCR** 机器人所需的组件清单

| 编号 | 大组件名称 | 需求/台 | 工时/台 | 小组件名称 | 编号 | 小组件/大组件 | 总需求 |
|---|---|---|---|---|---|---|---|
| | | | | 控制器 | A1 | 6 | 18 |
| A | 容器艇 | 3 | 3 | 划桨 | A2 | 8 | 24 |
| | | | | 感知器 | A3 | 2 | 6 |
| B | 机器臂 | 4 | 5 | 力臂组件 | B1 | 2 | 8 |
| | | | | 遥感器 | B2 | 4 | 16 |
| | | | | 蓄电池 | C1 | 8 | 40 |
| C | 动力系统 | 5 | 5 | 微型发电机 | C2 | 2 | 10 |
| | | | | 发电螺旋 | C3 | 12 | 60 |

为了顺利生产 WPCR，工厂开始生产组件时需支付生产准备费用。若当天下班组件有库存，则需要支付库存费用。此外为保障信誉，所有订单必须按期交付，不能缺货。

## 1.2 问题提出

**问题一:** 该工厂周一没有任何组件库存，周日结束后也不能留有组件库存。当天采购组件可立即组装，组件也可立即整合成 WPCR。请制定每周的生产计划使总成本最小。

**问题二:** 实际上, 部件需要提前一天入库才能给次日生产使用。在多周饱和生产时，需要统筹部件, 前一天备好次日所需部件。每周的 WPCR 需求和关键设备工时限制以及每次生产准备费用和单件库存费用数据见表 5、表 6，请据此制定每周生产计划使成本最低。

**问题三:** 承问题二, 为保障生产持续性, 工厂每 30 周 (210 天) 里需设置 7 次停工检修，每次检修为期 1 天。检修后关键设备第一天部件总工时限制将放宽 10%，随后逐日递减 2% 直至为 0。检修当日不能生产, 其订单应提前。检修间隔需 6 天以上，其余条件不变, 请设计总成本最小的检修计划并满足附录 2 中 30 周的 WPCR 外部需求。

**问题四:** 生产实际中，WPCR 订单数无法获悉，承问题二，给出表 7 为历史周订单数据，制定符合实际的最低成本的周生产计划, 既能使每日订单正常交付率超 95%，也能使整周订单正常交付率高于 85%。

## 二、 模型的假设

1. 假设问题一中, 小组件的采购不计工时, 大组件组装成 WPCR 机器人也不计工时。

2. 假设生产准备成本生产数量与无关的固定成本, 库存费用与库存数量成正比。

3. 假设问题三中, 每周的“关键设备工时限制”都与附表 5 的数据相同, 且每次生产准备费用和单件库存费用数据不变，任意两次检修之间要相隔 6 天以上。

4. 假设问题一、二、三中的 WPCR 机器人外部需求订单数量都是已知的。

## 三、 符号说明

| 符号 | 意义 | 单位 |
|---|---|---|
| Cost、CostT、CostP | 生产总费用、库存费用、生产准备费用 | ￥ |
| CostA、CostB、CostC | A、B、C 组件各自的总库存费用 | ￥ |
| $\Phi$、$\Psi$ | 组件单次生产准备费用、组件单件库存费用 | ￥ |
| $\Omega$、Q | 每台 WPCR 机器人各组件数量、外部需求订单数 | 件 |
| Lmt | 工时限制向量 | 时 |
| P | 产品产量矩阵 | 套 |
| M | 停工日次标识向量 | rad |
| $\gamma$、$\sigma$ | 量子比特概率幅 | rad |
| Pax、$\mu$ | 状态转移概率矩阵、状态概率 | rad |

# 四、 问题分析

## 4.1 考虑组件工时限制与组件库存清空使总成本最小

工厂为了提高直接经济效益, 总需从产量和库存的角度统筹生产总成本,WPCR 机器人生产也同样需要平衡外部需求与产量。通过题目假设, 本题理想化了小部件的采购流程、整合 A、B、C 部件的产出耗时, 同时限制周末结束生产时不留有部件库存, 以保证产品部件的利用率最高。由题意可知, 机器人生产过程的成本主要包括: 生产准备成本和库存费用两部分。而本题又要求保证每日订单交付不能缺货, 那么工作周内每一天都必须进行生产, 并且生产准备成本与机器人产量无关, 可以视作是一笔固定支出。以上可知, 尽可能降低总成本需要重点从降低部件的库存成本切入。基于题目所给的每天 WPCR 需求和关键设备工时限制数据 (见表 2), 我们考虑给出约束:1. 部件生产有总工时限制;2. 机器人产出至少应满足当天的外部需求;3. 周日结束生产时部件库存为 0。

表 **2**　每天 **WPCR** 需求和关键设备工时限制

| 天 | 周一 | 周二 | 周三 | 周四 | 周五 | 周六 | 周日 |
|---|---|---|---|---|---|---|---|
| WPCR 需求（个） | 39 | 36 | 38 | 40 | 37 | 33 | 40 |
| A、B、C 生产总工时限制（工时） | 4500 | 2500 | 2750 | 2100 | 2500 | 2750 | 1500 |

## 4.2 考虑组件生产入库的统筹实际

实际生产过程中, 部件采购以及生产整合各环节之间并不能无缝衔接, 故工厂统一规定当天组装的部件需提前入库, 次日才能用于进一步生产。规定给部件的供应施加周期性, 在应对外部需求订单时应该在前一天至少备好次日订单数对应所需的小部件库存。题目给出表 3“每次生产准备费用和单件库存费用数据”, 沿用表 2 中的工时限制与外部需求数据, 可以说明: **将小部件组装成大部件后的库存费用远远低于直接存储小部件**。

## 4.3 考虑通过关键设备维护检修提升生产效能

承接问题二的条件, 机器人产出效率受限于部件组装的工时限制。本题给出新方法: 工厂可以通过对关键设备停工检修以减轻工时受限, 提高产量。同时, 每日的外部需求订单也应该保证能够交付, 说明停工期间的订单货物需要提前备好, 以免出现缺货。

表 **3**　每次生产准备费用和单件库存费用

| 产品 | WPCR | A | A1 | A2 | A3 | B | B1 | B2 | C | C1 | C2 | C3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 生产准备费用 | 240 | 120 | 40 | 60 | 50 | 160 | 80 | 100 | 180 | 60 | 40 | 70 |
| 单件库存费用 | 5 | 2 | 5 | 3 | 6 | 1.5 | 4 | 5 | 1.7 | 3 | 2 | 3 |

### 4.4 考虑使未知需求订单交付尽量稳妥

生产环节中, 外部需求订单由于资金、物流等因素与计划数量存在波动, 不可预测。但是, 为了使企业信誉不受订单波动波及, 在实际的生产安排时，需要预备一定组件库存, 又得兼顾成本考量。据此推理, 安排生产前需要对以往的产销数据进行分析, 建立可以准确预测实际外部需求订单数的模型, 推演需求订单的波动情况, 科学指导 WPCR 机器人的生产实践。

## 五、 数据预处理

### 5.1 变量定义约定

为了便利后续模型的数学推理, 我们首先对题目中给出的各项数据给出符号约定; 同时为了便与计算机矩阵预算, 也将所给表格用向量的形式做了别名定义:

1. 设表 2 所示某生产周 WPCR 外部需求订单数为 $Q$, 其中周一到周日的外部需求订单数为 $q_1 - q_7$:

$$\begin{aligned} Q &= \begin{bmatrix} q_1, & q_2, & q_3, & q_4, & q_5, & q_6, & q_7 \end{bmatrix} \\ &= \begin{bmatrix} 39, & 36, & 38, & 40, & 37, & 33, & 40 \end{bmatrix} \end{aligned} \tag{5-1-1}$$

2. 设表 2 所示某生产周的工时限制为 $Lmt$, 其中周一到周日的单日公式限制为 $t_1 - t7$:

$$\begin{aligned} Lmt &= \begin{bmatrix} t_1, & t_2, & t_3, & t_4, & t_5, & t_6, & t_7 \end{bmatrix} \\ &= \begin{bmatrix} 4500, & 2500, & 2750, & 2100, & 2500, & 2750, & 1500 \end{bmatrix} \end{aligned} \tag{5-1-2}$$

3. 设造一台 WPCR 机器人所需要的各组件数量为行向量:

$$\begin{aligned} \Omega &= \begin{bmatrix} W, & A, & A_1, & A_2, & A_3, & B, & B_1, & B_2, & C, & C_1, & C_2, & C_3 \end{bmatrix} \\ &= \begin{bmatrix} 1, & 3, & 18, & 24, & 6, & 4, & 8, & 16, & 5, & 40, & 10, & 60 \end{bmatrix} \end{aligned} \tag{5-1-3}$$

4. 将表 3 中 12 种组件的“每次生产准备费用”设为行向量 Φ:

$$\begin{aligned}\Phi &= \left[\begin{array}{cccccccccccc} W, & A, & A_1, & A_2, & A_3, & B, & B_1, & B_2, & C, & C_1, & C_2, & C_3 \end{array}\right] \\ &= \left[\begin{array}{cccccccccccc} 240, & 120, & 40, & 60, & 50, & 160, & 80, & 100, & 180, & 60, & 40, & 70 \end{array}\right]\end{aligned} \tag{5-1-4}$$

5. 将表 3 中 12 种组件的“单件库存费用”设为行向量 Ψ:

$$\begin{aligned}\Psi &= \left[\begin{array}{cccccccccccc} W, & A, & A_1, & A_2, & A_3, & B, & B_1, & B_2, & C, & C_1, & C_2, & C_3 \end{array}\right] \\ &= \left[\begin{array}{cccccccccccc} 5, & 2, & 5, & 3, & 6, & 1.5, & 4, & 5, & 1.7, & 3, & 2, & 3 \end{array}\right]\end{aligned} \tag{5-1-5}$$

# 六、 问题一的模型

## 6.1 考虑模型建立

问题一将生产计划的时间尺度限制在一周之内, 规划周期较短; 同时要求, 一个生产周期内, 起止节点的组件库存都为空。从表 1 组装一台 WPCR 机器人所需产品部件清单中, 可知要想在结束生产时清空库存, 采购零部件 A1、A2、A3、B1、B2、C1、C2、C3 的数量应该恰好满足:

$$A1 : A2 : A3 : B1 : B2 : C1 : C2 : C3 = 18 : 24 : 6 : 8 : 16 : 40 : 10 : 60 \tag{6-1-1}$$

由题意可悉, 生产一台 WPCR 机器人的成本包括两方面: 生产准备费用和库存费用。

**1. 生产准备费用**

在问题分析中, 我们论述了: 对于问题一而言, 因为当周外部需求订单数可测, 可以准确的计算出所需的各种部件数量, **“生产准备费用是一项固定支出费用”**。在下文中将生产准备费用设为 $CostP$, 根据表 3 中给出的各部件的生产准备费用乘以所需数量求和即可求出。

**2. 库存费用**

从表 3 中给出了的各种部件的单间库存费用数据中, 不难得出不等关系:

$$\begin{aligned}\Psi_A &< \Psi_{A1} + \Psi_{A2} + \Psi_{A3} \\ \Psi_B &< \Psi_{B1} + \Psi_{B2} \\ \Psi_C &< \Psi_{C1} + \Psi_{C2} + \Psi_{C3} \\ \Psi_W &< \Psi_A + \Psi_B + \Psi_C\end{aligned} \tag{6-1-2}$$

从上述式子可以直观的得出结论, 要想尽量的降低库存费用, 就必须优先将部件整合成更高级的产品。不论产品能否当天售完, **以加工组装后的产品入库储存可以最大限度的降低库存费用**, 即尽量都在**当天将部件组装成 WPCR 机器人**。

**3. 模型的特征**

题目要求我们设计一种**成本最低**的生产方案, 给定了一个最优目标, 同时属于经典的方案规划类问题。通过观察题目中生产成本和各组件产量之间的特征, 我们发现:
各组件的生产成本都满足如下的**线性关系**:

组件成本 = 组件库存 × 组件单件库存成本 + 同类组件固定的生产准备费用
故我们考虑建立一个多变量**整数线性规划模型**

## 6.2 模型建立

### 6.2.1. 产品产量矩阵 $P$

通过观察题目要求解算结果表格的形式, 我们结合 MATLAB 语言运算的优势特点, 将附录 B 中附表 1 单元格 B2-E8 用矩阵的形式存储, 设计了如 6-2-1 所示的产品产量矩阵 $P$——矩阵 $P$ 中, 第一行到第七行表示从周一到周日 (7Day), 第一列到第四列依次表示 WPCR 机器人、A 组件、B 组件、C 组件的产量, 是一个 $7\times 4$ 的矩阵:

$$P=\left.\begin{array}{c} \overbrace{\begin{array}{cccc} W & A & B & C \end{array}} \\ \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{71} & \alpha_{72} & \alpha_{73} & \alpha_{74} \end{bmatrix} \end{array}\right\} Day \tag{6-2-1}$$

### 6.2.2. 线性规划模型的目标函数

依据题意, 我们需要把最优目标: **生产总成本最低**作为目标函数, 因此需要得出生产总成本的线性方程组。记总成本为产品产量矩阵 $P$ 的一个函数为 $Cost(P)$,WPCR 机器人、部件 A、B、C 的库存费用分别记为 $CostW$、$CostA$、$CostB$、$CostC$, 所有部件生产准备费用的总和记作 $CostP$, 易分析得总成本等于部件 A、B、C 的库存费用本与部件总生产准备费用之和:

$$
\begin{aligned}
CostP &= \sum_{i=1}^{7} q_i \cdot \sum_{j=1}^{12} \{\Omega_j \cdot \Phi_j\} \\
CostW &= \Psi 1 \cdot \sum_{i=1}^{7} \sum_{j=1}^{i} [\alpha_{(j,1)} - q_{(j,1)}] \\
CostA &= \Psi_2 \cdot \sum_{i=1}^{7} \sum_{j=1}^{i} [\alpha_{(j,2)} - 3 \cdot \alpha_{(j,1)}] \\
CostB &= \Psi_6 \cdot \sum_{i=1}^{7} \sum_{j=1}^{i} [\alpha_{(j,3)} - 4 \cdot \alpha_{(j,1)}] \\
CostC &= \Psi_9 \cdot \sum_{i=1}^{7} \sum_{j=1}^{i} [\alpha_{(j,4)} - 5 \cdot \alpha_{(j,1)}] \\
Cost(P) &= CostW + CostA + CostB + CostC + CostP
\end{aligned}
\tag{6-2-2}
$$

**6.2.3. 产量矩阵受到的不等式约束：**

(1) 组装 A、B、C 组件总耗时不能超过总工时限制 $Lmt$:

$$
Lmt_k \geqslant 3 \cdot \alpha_{(k,2)} + 5 \cdot \alpha_{(k,3)} + 5 \cdot \alpha_{(k,4)} \tag{6-2-3}
$$

(2) 每日 WPRC 机器人库存不应该低于当天的外部需求订单数 $Q$:

$$
Q_k \leqslant \sum_{i=1}^{k} \{\alpha_{(i,1)} - q_{(i,1)}\} \tag{6-2-3}
$$

(3) 每日 A、B、C 组件的库存足够用于当天的生产:

$$
\begin{aligned}
Q_k &\leqslant \frac{\sum_{i=1}^{k} \{\alpha_{(i,2)} - 3 \cdot \alpha_{(i,1)}\}}{3} \\
Q_k &\leqslant \frac{\sum_{i=1}^{k} \{\alpha_{(i,3)} - 4 \cdot \alpha_{(i,1)}\}}{4} \\
Q_k &\leqslant \frac{\sum_{i=1}^{k} \{\alpha_{(i,4)} - 5 \cdot \alpha_{(i,1)}\}}{5}
\end{aligned}
\tag{6-2-4}
$$

(4) 每日 A,B,C 部件的库存不应该少于满足当天机器人产量的需求:

$$
\begin{aligned}
\sum_{i=1}^{k} P_{(1,i)} &\leqslant \frac{\sum_{i=1}^{k} P_{(2,i)}}{3} \\
\sum_{i=1}^{k} P_{(1,i)} &\leqslant \frac{\sum_{i=1}^{k} P_{(3,i)}}{4} \\
\sum_{i=1}^{k} P_{(1,i)} &\leqslant \frac{\sum_{i=1}^{k} P_{(4,i)}}{5}
\end{aligned}
\tag{6-2-5}
$$

**6.2.4. 求解得出结论:**

基于上述分析, 在满足目标函数达到“总成本最低”时, 可以通过逻辑运算反解得出此时的产品产量矩阵, 该矩阵 $P_0$ 即实现最优目标时, 工厂的生产方案：

$$P_0 = argmin[Cost(P)] \tag{6-2-6}$$

设该周每天的生产准备费用 $CostP_i$、库存费用 $CostP_i$, 最终将产量矩阵 $P_0$ 带入线性公式可以得出使得成本最小的生产方案, 将本题最终结果填入附表 1 中, 见表 4:

$$\begin{aligned}
CostP_i &= P_{(i,1)} \cdot \sum_{j=1}^{12} \{\Omega_j \cdot \Phi_j\} \\
CostT_i &= CostW_i + CostA_i + CostB_i + CostC_i \\
&= \Psi_1 \cdot \sum_{i=1}^{i} [P_{(i,1)} - Q_i] + \Psi_2 \cdot \sum_{i=1}^{7} [P_{(i,2)} - 3 \cdot P_{(i,1)}] \\
&+ \Psi_6 \cdot \sum_{i=1}^{7} [P_{(i,3)} - 4 \cdot P_{(i,1)}] + \Psi_9 \cdot \sum_{i=1}^{7} [P_{(i,4)} - 5 \cdot P_{(i,1)}]
\end{aligned} \tag{6-2-7}$$

表 **4** 问题一的结果

| 日期 | WPCR 组装数量 | A 组装数量 | B 组装数量 | C 组装数量 | 生产准备费用 | 库存费用 |
|---|---|---|---|---|---|---|
| 周一 | 39 | 117 | 156 | 195 | 539760 | 0 |
| 周二 | 36 | 108 | 144 | 180 | 498240 | 0 |
| 周三 | 39 | 117 | 158 | 195 | 541200 | 5 |
| 周四 | 39 | 117 | 154 | 195 | 538320 | 0 |
| 周五 | 37 | 111 | 148 | 185 | 512080 | 0 |
| 周六 | 45 | 136 | 182 | 225 | 625180 | 60 |
| 周日 | 28 | 83 | 110 | 140 | 385140 | 0 |
| 总和 | 263 | 789 | 1052 | 1315 | 3639985 | |

# 七、问题二的模型

## 7.1 考虑模型建立

本题条件静载问题一的基础上新增了产品工艺流程限制, 更加符合生产实际。从题目要求来看, 本题的目标函数仍然是求最低成本生产方案, 生产成本仍然是生产准备费用和库存费用两部分。同时, 成本函数依然有强线性关系, 故我们考虑对问题一中的**整数线性规划模型**加以改进, 使之适用于新条件。

## 7.2 沿用线性规划模型改进

### 7.2.1. 新条件的适应改进

问题二在问题一的基础上创造了新的条件: 部件采购或组装完成后需要入库存储, 第二天才能用于下一步生产。沿用问题一的思路, 继续使用 6-2-1 中的产品产量矩阵 $P$ 作为求解对象, 在新条件下重新调整总成本函数的线性方程组。

### 7.2.2. 线性规划函数的目标函数

依据题意, 我们仍然需要把最优目标: **获最低生产总成本**作为目标函数, 可以对 6-2-2 的成本方程组进一步改进。记总成本为产品产量矩阵 $P$ 的一个函数为 $Cost(P)$,WPCR 机器人、部件 A、B、C 的库存费用分别即为 $CostW$、$CostA$、$CostB$、$CostC$, 所有部件生产准备费用的总和记作 $CostP$, 易分析得总成本等于部件 A、B、C 的库存费用本与部件总生产准备费用之和:

$$
\begin{aligned}
CostP &= \sum_{i=1}^{7} q_i \cdot \sum_{j=1}^{12} \{\Omega_j \cdot \Phi_j\} \\
CostW &= \Psi 1 \cdot \sum_{i=1}^{7} \sum_{j=1}^{i} [\alpha_{(j,1)} - q_{(j,1)}] \\
CostA &= \sum_{k=3}^{5} \{\Psi_k \cdot \Phi_k\} \cdot \sum_{i=1}^{7} \alpha_{(i,2)} + \Psi_2 \cdot \sum_{i=1}^{7} \{\sum_{j=1}^{i} [\alpha_{(j,2)} - 3 \cdot \alpha_{(j,1)}] + 3\alpha_{(1,1)}\} \\
CostB &= \sum_{k=7}^{8} \{\Psi_k \cdot \Phi_k\} \cdot \sum_{i=1}^{7} \alpha_{(i,3)} + \Psi_6 \cdot \sum_{i=1}^{7} \{\sum_{j=1}^{i} [\alpha_{(j,3)} - 4 \cdot \alpha_{(j,1)}] + 4\alpha_{(1,1)}\} \\
CostC &= \sum_{k=10}^{12} \{\Psi_k \cdot \Phi_k\} \cdot \sum_{i=1}^{7} \alpha_{(i,4)} + \Psi_9 \cdot \sum_{i=1}^{7} \{\sum_{j=1}^{i} [\alpha_{(j,4)} - 5 \cdot \alpha_{(j,1)}] + 5\alpha_{(1,1)}\} \\
Cost(P) &= CostW + CostA + CostB + CostC + CostP
\end{aligned}
\tag{7-2-1}
$$

本题中,A1、A2、A3、B1、B2、C1、C2、C3 组件采购后需入库, 第二天才能用于组装 A、B、C 组件。

### 7.2.3. 产量矩阵受到的不等式约束:

(1) 组装 A、B、C 组件总耗时不能超过总工时限制 $Lmt$:

$$Lmt_k \geqslant 3 \cdot \alpha_{(k,2)} + 5 \cdot \alpha_{(k,3)} + 5 \cdot \alpha_{(k,4)} \tag{7-2-2}$$

(2) 每日 WPRC 机器人库存不应该低于当天的外部需求订单数 $Q$:

$$Q_k \leqslant \sum_{i=1}^{k} \{\alpha_{(i,1)} - q_{(i,1)}\} \tag{7-2-3}$$

(3) 每日 A、B、C 组件的库存足够满足用于第二天的外部需求订单生产:

$$\begin{aligned} Q_k &\leqslant \frac{\sum_{i=1}^{k} \alpha_{(i,2)} - \sum_{j=2}^{k+1} 3 \cdot \alpha_{(j,1)}}{3} \\ Q_k &\leqslant \frac{\sum_{i=1}^{k} \alpha_{(i,3)} - \sum_{j=2}^{k+1} 4 \cdot \alpha_{(j,1)}}{4} \\ Q_k &\leqslant \frac{\sum_{i=1}^{k} \alpha_{(i,4)} - \sum_{j=2}^{k+1} 5 \cdot \alpha_{(j,1)}}{5} \end{aligned} \tag{7-2-4}$$

(4) 每日 A,B,C 部件的库存足以满足第二天实际机器人产量的组件需求:

$$\begin{aligned} \sum_{i=2}^{k} P_{(1,i)} &\leqslant \frac{\sum_{j=1}^{k-1} P_{(2,j)}}{3} \\ \sum_{i=2}^{k} P_{(1,i)} &\leqslant \frac{\sum_{j=1}^{k-1} P_{(3,j)}}{4} \\ \sum_{i=2}^{k} P_{(1,i)} &\leqslant \frac{\sum_{j=1}^{k-1} P_{(4,j)}}{5} \end{aligned} \tag{7-2-5}$$

**7.2.4. 求解得出结论:**

基于上述分析, 在满足目标函数达到“总成本最低”时, 可以通过逻辑运算反解得出此时的产品产量矩阵, 该矩阵 $P_1$ 即实现最优目标时, 工厂的生产方案:

$$P_1 = argmin[Cost(P)] \tag{7-2-6}$$

设该周每天的生产准备费用 $CostP_i$、库存费用 $CostP_i$, 最终将产量矩阵 $P_0$ 带入线性公式可以得出使得成本最小的生产方案, 将本题最终结果填入附表 1 中, 见表 5:

$$\begin{aligned} CostP_i &= P_{(i,1)} \cdot \sum_{j=1}^{12} \{\Omega_j \cdot \Phi_j\} \\ CostT_i &= CostW_i + CostA_i + CostB_i + CostC_i \\ &= \Psi_1 \cdot \sum_{i=1}^{i} [P_{(i,1)} - Q_i] + \Psi_2 \cdot \sum_{i=1}^{7} [P_{(i,2)} - 3 \cdot P_{(i,1)}] \\ &+ \Psi_6 \cdot \sum_{i=1}^{7} [P_{(i,3)} - 4 \cdot P_{(i,1)}] + \Psi_9 \cdot \sum_{i=1}^{7} [P_{(i,4)} - 5 \cdot P_{(i,1)}] \end{aligned} \tag{7-2-7}$$

表 **5**　问题二的结果

| 日期 | WPCR 组装数量 | A 组装数量 | B 组装数量 | C 组装数量 | 生产准备费用 | 库存费用 |
|---|---|---|---|---|---|---|
| 周一 | 39 | 117 | 156 | 195 | 504660 | 23479.5 |
| 周二 | 39 | 108 | 144 | 180 | 535180 | 25379 |
| 周三 | 36 | 117 | 158 | 195 | 538240 | 25321.5 |
| 周四 | 39 | 117 | 154 | 195 | 516040 | 24109.5 |
| 周五 | 39 | 111 | 148 | 185 | 545780 | 25604.5 |
| 周六 | 37 | 102 | 258 | 170 | 480980 | 22064 |
| 周日 | 34 | 117 | 34 | 195 | 519040 | 25369.5 |
| 总和 | 263 | 789 | 1052 | 1315 | 3811247.5 | |

# 八、 问题三的模型

## 8.1 考虑建立模型

本题承接问题二的条件, 在问题二的模型中, 条件约束数量已经较为庞大。本题新增的停工检修任务, 使得线性条件不再成立。同时, 附表 7 给出了 30 周 210 天需求数据, 数据量较大。综合来看, 问题三是一个有大量数据驱动的非线性优化问题。

对于非线性优化问题, 我们首先考虑使用智能算法解题。常见的智能算法有: 遗传算法、模拟退火、支持向量机等, 这些算法都有各自的优缺点。

同时, 本题背景决定了在 210 天里, 总会有一种确定的停工检修方案, 可以使得生产的总成本最低, 是一个又穷的优化问题。对于可解优化问题, 我们优先考虑使用遗传算法, 而**双链量子遗传算法 (DCQGA)** 是他的一种优秀的改进模型。DCQGA 算法在 “处理高维、复杂非线性函数组合优化问题，相比传统的遗传算法具有更优越的多样性”[1]。在量子遗传算法中，使用量子比特相位代表染色体, 用量子旋转门来操纵染色体的演变, 更加符合生物界随机演变的实际状态。

## 8.2 模型的建立

### 8.2.1. 产品产量矩阵 $P$

由附表 7 的数据形式, 我们已知 210 天的 WPCR 外部需求订单数量, 与问题二中提

供的某一周七天的 WPCR 外部需求数据相似。我们便考虑参考问题二中的“产品常量矩阵”$P$, 将 P 举证延长为 $210\times 4$ 的矩阵, 如 8-2-1 所示。第 1 行到第 210 行表示从第 1 天到第 210 天 (210Days), 第一列到第四列仍依次表示 WPRC 机器人、A 组件、B 组件、C 组件的产量:

$$P=\left.\overbrace{\begin{bmatrix} \alpha_{(1,1)} & \alpha_{(1,2)} & \alpha_{(1,3)} & \alpha_{(1,4)} \\ \alpha_{(2,1)} & \alpha_{(2,2)} & \alpha_{(2,3)} & \alpha_{(2,4)} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_{(210,1)} & \alpha_{(210,2)} & \alpha_{(210,3)} & \alpha_{(210,4)} \end{bmatrix}}^{W\qquad A\qquad B\qquad C}\right\}210Days \tag{8-2-1}$$

### 8.2.2. 双链量子遗传算法的优化目标函数

在 8.1 的分析中, 我们阐述了仍然以**获取最低总成本**为目标函数的原因, 在本题中, 由于总成本数量是一个仅仅与**产品产量矩阵** $P$ 相关的函数, 故可以沿用问题二中 7-2-1 中的目标函数进行修正:

如 8-2-1 所示, 由于附表 7 中给出的 210 天工作周期不会形成循环, 可以忽略第一天的机器人产量, 即从 $\alpha_{(2,1)}$ 开始计算, 故在 3、4、5 式少去了余项 $n\cdot\alpha_{(1,1)}$ 其中 WPCR 机器人、部件 A、B、C 的库存费用分别记为 $CostW$、$CostA$、$CostB$、$CostC$, 所有部件生产准备费用的总和记作 $CostP$, 易分析得总成本 $Cost(P)$ 等于部件 A、B、C 的库存费用本与部件总生产准备费用之和:

$$\begin{aligned} CostP &= \sum_{i=1}^{210} q_i\cdot\sum_{j=1}^{12}\{\Omega_j\cdot\Phi_j\} \\ CostW &= \Psi 1\cdot\sum_{i=1}^{210}\sum_{j=1}^{i}[\alpha_{(j,1)}-q_{(j,1)}] \\ CostA &= \sum_{k=3}^{5}\{\Psi_k\cdot\Phi_k\}\cdot\sum_{i=1}^{210}\alpha_{(i,2)}+\Psi_2\cdot\sum_{i=1}^{210}\sum_{j=1}^{i}[\alpha_{(j,2)}-3\cdot\alpha_{(j,1)}] \\ CostB &= \sum_{k=7}^{8}\{\Psi_k\cdot\Phi_k\}\cdot\sum_{i=1}^{210}\alpha_{(i,3)}+\Psi_6\cdot\sum_{i=1}^{210}\sum_{j=1}^{i}[\alpha_{(j,3)}-4\cdot\alpha_{(j,1)}] \\ CostC &= \sum_{k=10}^{12}\{\Psi_k\cdot\Phi_k\}\cdot\sum_{i=1}^{210}\alpha_{(i,4)}+\Psi_9\cdot\sum_{i=1}^{210}\sum_{j=1}^{i}[\alpha_{(j,4)}-5\cdot\alpha_{(j,1)}] \\ Cost(P) &= CostW+CostA+CostB+CostC+CostP \end{aligned} \tag{8-2-2}$$

在**遗传算法**中, **适应度函数**的选取应该是一个趋向于最优且可以收敛的函数; 在本题中, 恰好可以将**总成本函数**作为算法的适应度函数, 因为总成本函数在有限的时间区间内存在一个极值点, 总是会收敛在一个最低成本位置, 非常适合做迭代收敛的标度值。

**8.2.3. 停工日期标识向量 $M$**

为了便与后续算法中的描述, 需要初始化一个标识向量来记录停工日期的选取; 故我们定义了停工日期标识向量 $M$,$M$ 是一个 210 列的零向量, 列数表示天数, 用 0 表示正常开工, 用 1 表示停工:

$$\begin{aligned} M &= \begin{bmatrix} m_1, & m_2, & m_3, & m_4, & \ldots & m_{209}, & m_{210} \end{bmatrix} \\ &= \begin{bmatrix} 0, & 0, & 0, & 0, & \ldots & 0, & 0 \end{bmatrix} \end{aligned} \tag{8-2-3}$$

**8.2.4. 工时限制的规定**

本题中可以通过停工检修关键设备, 牺牲检修当天的生产能力, 使得后续 5 天工时限制依次放宽:10%、8%、6%、4%、2%, 直至第 6 天恢复正常的工时限制。同时, 本题题干假设: 附表 7 中 30 周数据中“每周的关键设备工时限制都与附表 5 的数据相同”。故可以有下述约定:

(1) 30 周的工时限制初始化都相同, 如表 6 所示, 同时将工时限制按第 1 天到第 210 天的顺序重新定义为向量 $Lmt$:

表 **6** 工时限制的初始矩阵

| 天 | 周一 | 周二 | 周三 | 周四 | 周五 | 周六 | 周日 |
|---|---|---|---|---|---|---|---|
| 第 1 周 | 4500 | 2500 | 2750 | 2100 | 2500 | 2750 | 1500 |
| 第 2 周 | 4500 | 2500 | 2750 | 2100 | 2500 | 2750 | 1500 |
| 第 ... 周 | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 第 15 周 | 4500 | 2500 | 2750 | 2100 | 2500 | 2750 | 1500 |
| 第 ... 周 | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 第 30 周 | 4500 | 2500 | 2750 | 2100 | 2500 | 2750 | 1500 |

$$Lmt = \begin{bmatrix} t_1, & t_2, & t_3, & t_4, & \ldots & t_{209}, & t_{210} \end{bmatrix} \tag{8-2-4}$$

(2) 当确定某一天需要进行停工检修时, 由于停工不能进行生产, 当天的工时限制应更新为 0, 并对停工向量 $M$ 做标记; 同时, 随后 5 天的工时限制依次更新为原来的基础上的:110%、108%、106%、104%、102%。

设第 i 天进行停工检修, 在 MATLAB 中可以定义一个赋值函数 $dec(i)$ 实现下述目的:

$$dec(i)=\begin{cases} t_i=0, \qquad m_i=1 \\ t_{i+1}=t_{i+1}[100+(12-2\times 1)]\% \\ t_{i+2}=t_{i+3}[100+(12-2\times 2)]\% \\ \quad\cdots\cdots \\ t_{i+5}=t_{i+5}[100+(12-2\times 5)]\% \end{cases} \tag{8-2-5}$$

**8.2.4. 部分向量更新**

在问题四中, 由于表 4 给出了连续 210 天的机器人外部需求订单数量, 需要重新定义需求向量 $Q$:

$$\begin{aligned} Q&=\begin{bmatrix} q_1, & q_2, & q_3, & q_4, & \ldots & q_{209}, & q_{210} \end{bmatrix} \\ &=\begin{bmatrix} 39, & 36, & 38, & 40, & \ldots & 32, & 44 \end{bmatrix} \end{aligned} \tag{8-2-6}$$

根据题目假设, 本题中 WPCR 机器人所需要的各组件数量 Ω、12 种组件的“每次生产准备费用”Φ、12 种组件的“单件库存费用”Ψ 皆与 5-1-3、5-1-4 和 5-1-5 中的定义保持一致, 沿用上述定义可不做修正。

**8.2.5. 产量矩阵受到的不等式约束**

(1) 每日组装 A、B、C 组件总耗时不能超过当天总工时限制 $Lmt$:

$$Lmt_k \geqslant 3\cdot\alpha_{(k,2)}+5\cdot\alpha_{(k,3)}+5\cdot\alpha_{(k,4)} \tag{8-2-7}$$

(2) 每日 WPRC 机器人库存不应该低于当天的外部需求订单数 $Q$:

$$Q_k \leqslant \sum_{i=1}^{k}\{\alpha_{(i,1)}-q_{(i,1)}\} \tag{8-2-8}$$

(3) 每日 A、B、C 组件的库存足够满足用于第二天的外部需求订单生产:

$$\begin{aligned} Q_k &\leqslant \frac{\sum_{i=1}^{k}\alpha_{(i,2)}-\sum_{j=2}^{k+1}3\cdot\alpha_{(j,1)}}{3} \\ Q_k &\leqslant \frac{\sum_{i=1}^{k}\alpha_{(i,3)}-\sum_{j=2}^{k+1}4\cdot\alpha_{(j,1)}}{4} \\ Q_k &\leqslant \frac{\sum_{i=1}^{k}\alpha_{(i,4)}-\sum_{j=2}^{k+1}5\cdot\alpha_{(j,1)}}{5} \end{aligned} \tag{8-2-9}$$

(4) 每日 A,B,C 部件的库存足以满足第二天实际机器人产量的组件需求:

$$\begin{gathered}\sum_{i=2}^{k} P_{(1,i)} \leqslant \frac{\sum_{j=1}^{k-1} P_{(2,j)}}{3} \\ \sum_{i=2}^{k} P_{(1,i)} \leqslant \frac{\sum_{j=1}^{k-1} P_{(3,j)}}{4} \\ \sum_{i=2}^{k} P_{(1,i)} \leqslant \frac{\sum_{j=1}^{k-1} P_{(4,j)}}{5}\end{gathered} \tag{8-2-9}$$

**8.2.5. 产量矩阵受到的等式约束**

(1)210 天 WPCR 机器人的总产量应该等于 210 天对机器人的总需求:

$$\sum_{i=1}^{210} P_{(1,i)} = \sum_{j=1}^{210} Q_{(j)} \tag{8-2-10}$$

**8.2.6. 双链量子遗传算法 (DCQGA) 的参数约定**

(1) 适应度函数:

遗传算法的适应度函数是迭代完成的信标, 当适应度函数趋于平稳时可以说明, 此时目标函数已经趋于最优, 故本题中以取得“最低成本”为优化目标”。

$$\begin{gathered}\beta = min\{cost(P)\} \\ fitness = min\left[\begin{array}{cccccc} \beta_1, & \beta_2, & \beta_3, & \ldots & \beta_{i-1}, & \beta_i \end{array}\right]\end{gathered} \tag{8-2-11}$$

当 $\beta_i = lim_{i \to \varepsilon} fitness$ 时, 说明此时优化目标已经收敛处于稳定态, 可以直观地从适应度曲线反映出。

(2) 染色体、个体:

“遗传算法的思想是模拟生物界的遗传过程：把问题的解用染色体表示, 在计算机中用字符串模拟, 经过多次迭代演变, 形成一个由具有不同染色体个体组成的种群。[2]”，。每一个体的染色体是作用于整个种群性质的最基本单元, 可以简化理解成“自变量”。在本题的模型中, 成本函数由产量矩阵决定, 而**产量矩阵受到停工检修的影响**, 故停工检修日期选择是最基础的因素, 故可以把**“每天是否在停工检修”的标识** $m_i$ 作为遗传算法的染色体, 将 7 次检修的安排即**停工日期标识向量** $M$ 作为影响最终总生产成本的个体。

(3) 繁殖：

GA 算法模拟自然界中的优胜劣汰, 适者生存。210 天中每天的生产成本受到染色体的作用, 有最优秀染色体组成的个体会存活到最后。由于本题中约束条件数量高达 1677 条, 为了尽可能地提高计算机迭代速度, 人工设定**只有 1 个种群**, 约定种群所在环境承载力最多只能容纳 **40 个个体**生存, 排位靠后的个体会被种群淘汰, 得出有优秀染色体排列的最优个体。

(4). 交叉：

为了尽量模拟自然演化, 对于一部分个体 $M$, 可以将两个体同一天的染色体 $m_i$ 做 0-1 对调, 进行个体交配, 随机交换染色体, 各自生成新的个体。在本模型中, 而个体是否发生交叉也是由量子旋转门的指向决定。

(5). 变异:

考虑到本题中有部分个体总是不发生交叉, 我们对连续多代没有变化的个体做变异处理, 让其总成本序列发生调动。而个体是否发生变异也是由一个量子旋转门最终指向决定。

**8.2.6.DCQGA 算法的演化迭代**

**(1) 量子比特编码**

在量子遗传算法中, 染色体常以量子位的概率幅进行编码，本题采用 8-2-12 量子比特编码：

$$Pass = \left[\begin{array}{c} \begin{vmatrix} cos(\theta_{i1}) \\ sin(\theta_{i1}) \end{vmatrix} \begin{vmatrix} cos(\theta_{i2}) \\ sin(\theta_{i3}) \end{vmatrix} \begin{matrix} \wedge \\ \wedge \end{matrix} \begin{vmatrix} cos(\theta_{ik}) \\ sin(\theta_{ik}) \end{vmatrix} \end{array}\right]_{\theta_{ij}=2\pi\times rand, i=1,2\cdots n, j1,2\cdots k} \tag{8-2-12}$$

量子比特编码可以解析为二进制编码形式——每个染色体 $(m)$ 可以解析成概率幅式, 假设一个染色体有 r 个基因片段组成, 第 i 个基因可以表示为 $\frac{\gamma_i}{\lambda_i}$ 表示, 每个染色体都可以对应成唯一的向量, 其中 $\gamma$ 和 $\lambda$ 叫做量子比特概率幅, 都是与染色体相关的复变量:

$$\left[\begin{array}{ccccc} \frac{\gamma_1}{\lambda_1} & \frac{\gamma_2}{\lambda_2} & \frac{\gamma_i}{\lambda_i} & \cdots & \frac{\gamma_n}{\lambda_n} \end{array}\right] \tag{8-2-13}$$

其中 $|\gamma|^2$ 表示 0 的概率, $|\lambda|^2$ 表示 1 的概率, 而且必须满足在同一个单位元上:

$$|\gamma|^2 + |\lambda|^2 = 1 \tag{8-2-14}$$

**(2) 初始化种群随机编码:**

在进行迭代演化之前, 需要将上文中定义的染色体种群进行初始化, 即使用 8-2-12 的编码方式在一个圆周 (0,2$\pi$) 内随机生成 40 个向量 $M$ 组成一个初始种群。量子染色体的适应度值为 $\theta$ 在投影指数函数的数值。

**(3) 解空间变换:**

种群中的每条染色体在量子比特中都有 2n 的概率振幅。利用线性变换将 2n 概率振幅映射到优化目标 $min\{min[cost(P)]\}$ 的解空间上。

**(4) 迭代演化:**

运用量子旋转门改变量子比特相位，染色体的量子比特相位可以解析为的二进制字符串, 每一次量子旋转都会更新量子位的概率幅，当对应的两个个体发生耦合旋转时, 达到基因交叉的效果; 单个个体自旋时发生染色体变异。

**(5) 量子旋转门更新:**

在 DCQGA 中，量子旋转门用于更新量子比特相位。量子旋转门的定义为:

在量子遗传中, 由于 $|\gamma|^2+|\lambda|^2=1$, 当量子比特发生旋转 Δθ 时, 概率幅将由 $\frac{\gamma}{\lambda}$ 变换为 $\frac{\gamma'}{\lambda'}$ 其中 Δθ 为旋转角度，更新过程可表示为:

$$\begin{bmatrix}\gamma_i'\\ \lambda_i'\end{bmatrix}=\begin{bmatrix}cos(\Delta\theta_i) & -sin(\Delta\theta_i)\\ sin(\Delta\theta_i) & cos(\Delta\theta_i)\end{bmatrix}\cdot\begin{bmatrix}\gamma_i\\ \lambda_i\end{bmatrix} \tag{8-2-15}$$

**(6) 适应度收敛:**

每一次的更新量子旋转门之后, 都要对该个体所对应的**生产总成本**做特征分析。若成本高于前四十位, 该个体会被淘汰。每一次遗传迭代总会使得唯一种群的排序梯度发生改变, 最终前四十个个体的成本回非常接近, 且都趋于最低成本, 此时童通过适应度曲线可以判断出收敛优度。当量子遗传收敛后, 可以认为种群中排序第一位的最优个体所对应的**生产总成本最低**。

## 8.3 求解得出结果

**8.3.1.DCQGA 算法迭代结果**基于上述的双链量子遗传算法、优化目标函数定义与大量的约束, 我们通过 MATLAB 编程, 在上述算法思想实现成为逻辑代码, 设置遗传迭代次数为 100 次, 经过计算机的演算进化, 我们最终得出特性最优的**停工日期标识向量** $M_0$, 停工信标 $m=1$ 的天数分别如表 7 所示, 同时将产量矩阵 $P$ 回带式 8-2-2 得出最优方案的总成本为 113819713.3￥:

表 **7** 问题三的结果

| 第 1 次 | 第 2 次 | 第 3 次 | 第 4 次 | 第 5 次 | 第 6 次 | 第 7 次 | 总成本 |
|---|---|---|---|---|---|---|---|
| 2 | 18 | 50 | 102 | 146 | 200 | 206 | 113819713.3 |

**8.3.2.DCQGA 算法迭代适应度曲线**

我们在遗传算法迭代过程记录了各次迭代的适应度函数值, 绘制如下图 1 的适应度曲线:

通过本题中双链遗传算法的适应度曲线可以看出, 模型在迭代小于第 30 代时就已经收敛于最优, 说明本模型具有较优秀的模型适应度。

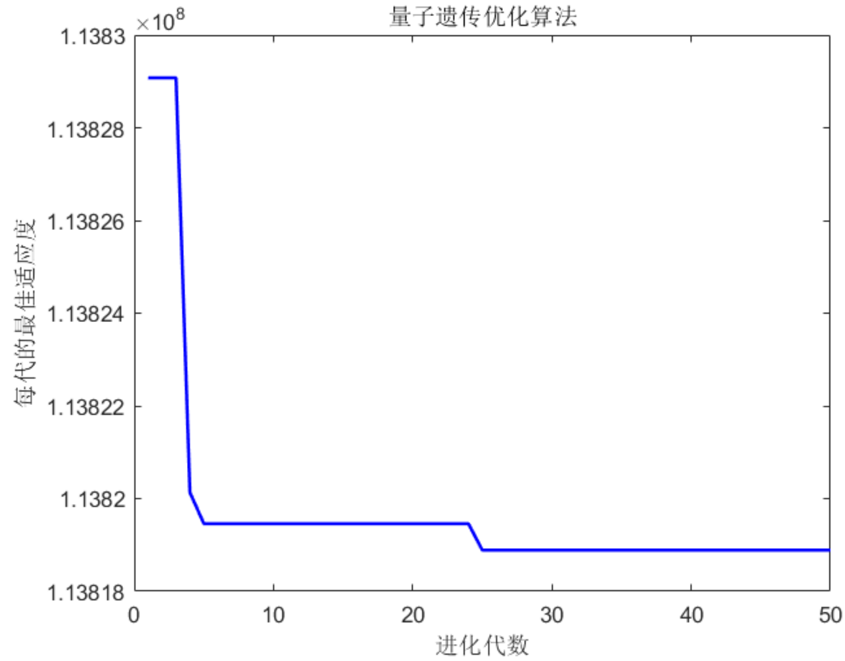


图 **1**　双链量子遗传算法的迭代适应度曲线

# 九、 问题四的模型

## 9.1 考虑建立模型

本题题干新增约束, 使未来工作周的外部需求订单数量是未知的; 同时, 通过观察附表 7 中的数据形式, 我们认为: 所给 30 周订单数据在周一的外部需求拥有一定的相关性; 周二到周日每列数据都是各自相关的。故需要分别对每列数据做分析, 建立出未来一周的订单预测模型才能更符合实际。

## 9.2 模型的建立

### 9.2.1. 马尔可夫外部需求概率预测:

**(1) 部分参数约定**我们将附表 7 中的数据整理成一个 $30 \times 7$ 的列向量矩阵, 依次用 $\sigma_1$ 、$\sigma_2$ 、$\sigma_3$ 、$\sigma_4$ 、$\sigma_5$ 、$\sigma_6$ 、$\sigma_7$ 表示周一到周日的各列的外部需求订单数, 即:

$$
\begin{aligned}
Q &= \begin{bmatrix} \sigma_1, & \sigma_2, & \sigma_3, & \sigma_4, & \sigma_5, & \sigma_6, & \sigma_7 \end{bmatrix} \\
&= \left.\begin{bmatrix} 39 & 36 & 38 & 40 & 37 & 33 & 40 \\ 39 & 33 & 37 & 43 & 34 & 30 & 39 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 37 & 41 & 39 & 41 & 36 & 32 & 44 \end{bmatrix}\right\} 30rows
\end{aligned} \tag{9-2-1}
$$

分别对 7 个列向量 $\sigma_i$ 分析特征, 建立各自对未来一天的短期预测模型 0。

**(2) 转移状态概率矩阵**

转移概率矩阵是由出现所有状态间的转移概率构成的矩阵。转移概率是马尔科夫链中概念, 由于本题中数据较多, 且计算重复累赘, 为了尽量缩短预算时间, 本题中采用**单步转移概率**来构造转移概率矩阵。

如下需对 7 个列向量各自计算转移概率并构造矩阵, 以 $\sigma_1$ 为例, 周一序列中出现 **11 种情况**, 分别是:31、33、35、36、37、38、39、40、41、42、43。我们设每一种情况依次为:$d_{(1)}$-$d_{(11)}$; 即在 7 组序列中用 $d_{(i)}$ 表示出现的不同状态。以 $\sigma_1$ 为例, 依次对 7 个订单需求列向量建立状态 $d_{(i)}$ 的对照字典:

$$\sigma_1^T = \begin{matrix} [d_{(7)}:39 & d_{(7)}:39 & d_{(10)}:42 & d_{(6)}:38 & d_{(6)}:38 \\ d_{(8)}:40 & d_{(9)}:41 & d_{(2)}:33 & d_{(3)}:35 & d_{(11)}:43 \\ d_{(6)}:38 & d_{(5)}:37 & d_{(6)}:38 & d_{(7)}:39 & d_{(8)}:40 \\ d_{(3)}:35 & d_{(11)}:43 & d_{(3)}:35 & d_{(4)}:36 & d_{(5)}:37 \\ d_{(5)}:37 & d_{(7)}:39 & d_{(8)}:40 & d_{(6)}:38 & d_{(6)}:38 \\ d_{(1)}:31 & d_{(8)}:40 & d_{(2)}:33 & d_{(3)}:35 & d_{(5)}:37\ ] \end{matrix} \tag{9-2-2}$$

单步转移概率的数学定义为: 在非负序列中中, 出现**相邻 (单步) 相同状态转移情况的次数与状态 A 为起点全体状态转移次数以的比值**。

$$P_{(m,m+n)} = P(d_m \rightarrow d_{m+n}) = P(d_m \mid d_{m+n}) \tag{9-2-3}$$

转移概率矩阵是一个 $i \times i$ 的方阵, 矩阵维数由状态数量决定:

$$Pax = \begin{bmatrix} P_{(1,1)} & P_{(1,2)} & \cdots & P_{(1,i-1)} & P_{(1,i)} \\ P_{(2,1)} & P_{(2,2)} & \cdots & P_{(2,i-1)} & P_{(2,i)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ P_{(i-1,1)} & P_{(i-1,2)} & \cdots & P_{(i-1,i-1)} & P_{(i-1,i)} \\ P_{(i,1)} & P_{(i,2)} & \cdots & P_{(i,i-1)} & P_{(i,i)} \end{bmatrix} \tag{9-2-4}$$

对照序列 9-2-2, 根据单步转移概率定义, 可以得出 7 个转移概率矩阵 $Pax_1$-$Pax_7$。

**(3) 状态概率**

$$\mu_j(n) = \sum_{i=1}^{k} \mu_i(n-1) P_{(i,j)} \qquad (j = 1, 2, \cdots k) \tag{9-2-5}$$

基于上述式子可以得出马尔科夫预测模型的概率递推公式:

$$\begin{cases} \mu(1) = \mu(0) \cdot P \\ \mu(2) = \mu(1) \cdot P = \mu(0) \cdot P^2 \\ \qquad \vdots \\ \mu(n) = \mu(n-1) \cdot P = \mu(0) \cdot P^k \end{cases} \tag{9-2-6}$$

经过计算机程序演算, 可以得出附表 7 中每周分别发生 11 状态各自的概率, 由于篇幅此处不以赘述。

**(4) 终极状态概率预测**

终极状态概率又叫“平衡状态概率”, 是将优先次状态转移推广至无穷, 经过无穷多次状态转移后所得到的状态概率会趋于稳定, 称为终极状态概率。此处将终极状态概率记作向量 $\mu$=[$\mu_1$,$\mu_2$,$\cdots$,$\mu_n$], 则:

$$\mu_i = \lim_{n\to\infty} \mu_n \tag{9-2-7}$$

将式 9-2-7 各项带入 9-2-6 关于状态概率的递推公式, 可以求出, 周一发生状态 $d_{(1)}$-$d_{(11)}$ 的终极概率向量 $\mu^{(1)}$, 且 $\mu^{(n)}$ 满足下述约束 ($n = 1, 2, \cdots, 7$):

$$\begin{cases} \mu^{(n)} = \mu \cdot Pax \\ 0 \le \mu(i) \le 1 \qquad (i = 1, 2, \cdots n) \\ \sum_{i=1}^{k} \mu(i) = 1 \end{cases} \tag{9-2-8}$$

经过大量的程序计算, 最终得出了周一到周日的马尔科夫状态概率, 如下表 8 所示:

表 **8**　周一至周日所有可能发生的状态概率表

| 状态 | d(1) | d(2) | d(3) | d(4) | d(5) | d(6) | d(7) | d(8) | d(9) | d(10) | d(11) | d(12) | d(13) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 周一需求 | 31 | 33 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | - | - |
| $P_{d(i)}$ | 0.035971 | 0.064748 | 0.129496 | 0.032374 | 0.151079 | 0.215827 | 0.115108 | 0.129496 | 0.032374 | 0.028777 | 0.064748 | - | - |
| 周二需求 | 28 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 |
| $P_{d(i)}$ | 0.032460 | 0.032460 | 0.064919 | 0.073790 | 0.062097 | 0.033871 | 0.068952 | 0.266734 | 0.182460 | 0.073790 | 0.039315 | 0.032460 | 0.036694 |
| 周三需求 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 44 | - | - |
| $P_{d(i)}$ | 0.033145 | 0.029872 | 0.068418 | 0.103855 | 0.146984 | 0.132990 | 0.103118 | 0.159096 | 0.099435 | 0.063344 | 0.059743 | - | - |
| 周四需求 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 44 | - | - |
| $P_{d(i)}$ | 0.033145 | 0.029872 | 0.068418 | 0.103855 | 0.146984 | 0.132990 | 0.103118 | 0.159096 | 0.099435 | 0.063344 | 0.059743 | - | - |
| 周五需求 | 31 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 42 | - | - | - | - |
| $P_{d(i)}$ | 0.034535 | 0.141678 | 0.138141 | 0.131460 | 0.073394 | 0.129495 | 0.092258 | 0.194292 | 0.064747 | - | - | - | - |
| 周六需求 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| $P_{d(i)}$ | 0.058890 | 0.065685 | 0.091733 | 0.027180 | 0.072480 | 0.081540 | 0.095130 | 0.117780 | 0.144960 | 0.081540 | 0.101925 | 0.031710 | 0.029445 |
| 周日需求 | 33 | 34 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | - |
| $P_{d(i)}$ | 0.034602 | 0.034602 | 0.035986 | 0.034602 | 0.103806 | 0.207612 | 0.131488 | 0.143945 | 0.035294 | 0.105882 | 0.096886 | 0.035294 | - |

### 9.2.2. 计算机模拟测试集

根据表 8 中周一至周日的马尔科夫状态概率, 可以分别对第 31 周的周一、周二到周日, 分别进行概率模拟。

为了使得计算机数据模拟尽量趋于平均水平, 我们根据表 8 依次生成了 100000 个测试周的外部需求周订单预测作为校验正常交付率的测试集。

**9.2.3. 测试集检验测筛**

由于通过马尔科夫概率模拟出的测试集元素, 各周外部需求是相互独立的, 我们依次用模拟周外部需求订单数量, 更新了 5-1-1 中的外部需求订单数量向量 $Q$:

$$Q^{(k)}=\left[q_1^{(k)},q_4^{(k)},q_3^{(k)},q_5^{(k)},q_5^{(k)},q_6^{(k)},q_7^{(k)}\right]\qquad(k=1,2\cdots,100000)\tag{9-2-9}$$

将模拟出的 10000 中产品外部需求情况, 回带问题二中**章节 7.2 中的最低成本生产方案规划模型**, 重复 100000 次多变量线性规划, 最中得出了这 100000 个模拟需求订单所对应的**产品产量矩阵** $P$ 和最低总成本 $Cost(P)$, 由于数据集过于庞大, 将以附件形式存放在支撑材料包中, 见“附件：表 1-问题四中 100000 例计算机模拟数据及总成本结果”。

**9.2.4. 测试集正常交付率排序**

通过比较 100000 例模拟数据在各自的**产品产量矩阵** $P^{(k)}$ 中 $\alpha^{(k)}{}_{(1,1)}$-$\alpha^{(k)}{}_{(7,1)}$(即模拟该周每天的机器人产量), 与模拟外部需求订单数据 $Q^{(k)}$, 依次比较即可求出每天的正常交付率。

然后将 100000 例模拟测试集中满足: **每日订单正常交付率大于 95%，整周订单正常交付率大于 85%** 对应的所有产量矩阵 $P$ 取出整理进一个新数据库。最终对这些满足要求的生产计划, 按照各自在 9.2.3 计算出的最低总成本 $Cost(P)$ 在 EXCEL 中升序排列, 排序第一位的产量矩阵就可以认为对应着**最稳妥的生产计划**, 如表 9 所示。

表 **9**　排序第一的模拟订单数

| 日期 | 星期一 | 星期二 | 星期三 | 星期四 | 星期五 | 星期六 | 星期天 |
|---|---|---|---|---|---|---|---|
| 模拟需求订单数 | 40 | 39 | 35 | 43 | 38 | 29 | 39 |

**9 . 3 求解得出结果**

基于 9.2 中叙述的模型设计，我们在 MATLAB 中编写了马尔科夫预测函数, 模拟出马尔科夫状态概率后进行大量的计算机模拟, 最终得出满足问题四“最稳妥”条件的产品产量生产方案, 如表 10 所示:

*** 结论注释:** 本题题目中叙述, 在未知未来周订单的前提下, 制定最稳妥的周生产计划。通过校验表模拟概率: 我们发现, 出现表 9 中最优订单计划的**状态概率仅为 0.000011**, 是一个极小的概率事件。不能用极特殊的需求订单数量特例来概括未来七天的“WPCR 外部需求”, 故我们对表 10 中“WPCR 外部需求”一列留空。

表 **10**　问题四的结果

| 日期 | WPCR 外部需求数量 | WPCR 组装数量 | A 组装数量 | B 组装数量 | C 组装数量 | 生产准备费用 | 库存费用 |
|---|---|---|---|---|---|---|---|
| 周一 | —— | 43 | 129 | 172 | 215 | 560020 | 26081.5 |
| 周二 | —— | 43 | 120 | 160 | 200 | 554320 | 26035 |
| 周三 | —— | 40 | 120 | 160 | 200 | 530200 | 24775 |
| 周四 | —— | 40 | 114 | 152 | 190 | 549800 | 26004 |
| 周五 | —— | 38 | 120 | 160 | 200 | 540180 | 25221 |
| 周六 | —— | 40 | 109 | 198 | 182 | 439360 | 19943.4 |
| 周日 | —— | 26 | 98 | 78 | 163 | 562920 | 27971.5 |
| 总和 | —— | 270 | 810 | 1080 | 1350 | 3912831.4 | |

# 十、 模型的评价改进

## 10 . 1 模型的评价

### 10.1.1. 线性规划模型的优点和不足:

优点：

1. 线性规划模型是一种常见的可穷规划算法, 原理简单, 便于模型设计;

2. 有统一的模型算法，对于线性规划问题普遍使用，是一种可以解决多变量最优问题的决策方法。

不足：

1. 对于数据的准确性要求高，而且只能对线性的问题进行约束优化;

2. 当自变量维数较多时, 计算量过大，对计算机性能占用极大。

### 10.1.2. 量子遗传算法的优点和不足:

优点:

1. 对比传统的遗传算法, 搜索范围更广，有更高的模型适应度，相比之下迭代更高效，优化效果更好, 而且便于最优检索。

2. 兼有遗传算法的智能特性, 对于非线性问题有优秀的求解能力。

不足: 在本题模型中, 量子遗传算法在少于 30 次迭代时就已经收敛了, 出现了早熟收敛的问题, 而且可能出现了局部最优问题。

### 10.1.3. 马尔科夫预测的优点和不足:

优点：

马尔科夫预测模型是一种适用于随机现象的数学模型，与历史状态没有直接联系。该模型对过程的状态预测有不错的效果，如第 7 章中论述的, 总成本函数与前一天的产

量有关, 是一个离散过程。而马氏预测适用于本题中订单“日正常交付率和周正常交付率”的过程预测。

不足:

1. 马尔科夫预测对于短期的预测有一定可信度, 对于中长期预测误差过大。

2. 马尔科夫预测是基于随机概率的预测, 对于如本题中预测状态较多的情况, 每个事件发生概率都偏小, 即对结果的预测效果不佳。

### 10.2 模型的改进

#### 10.2.1. 问题一模型的改进:

针对问题一中线性规划模型数据精度要求过高的问题, 可以使用模糊线性规划[3]模型作进一步改进; 模糊线性规划将线性约束的边界放宽, 运用不等约束取代等式约束, 放宽了数据的精度限制。

#### 10.2.2. 问题二模型的改进:

针对问题二中线性规划模型变量维数、条件约束过多导致的计算量过大的问题:
可以通过改进求解线性规划算法提升性能,MATLAB 中内置的解线性规划函数 intlinprog() 使用了传统的二分法和黄金分割法, 计算性能偏弱。后续研究可以运用“切割面求优算法”[4]提升计算的速度。

#### 10.2.3. 问题三模型的改进:

针对本题中 DCQGA 双链量子遗传算法过早收敛的问题，在后续的研究中可以考虑, 使用“IQGA 算法”对量子遗传算法加以改进。“IQGA 算法可以有效避免传统量子遗传算法早熟收敛, 具有寻优能力强、收敛快等优点, 适合于一般连续函数优化问题”[5]。

#### 10.2.4. 问题四模型的改进:、

针对对本题中马尔科夫预测对结果预测的效果不佳的问题：可以结合多元回归的思想, 将模型改进为“多元回归的马尔科夫链模型”[6], 也可运用“隐马尔科夫模型与 EM 算法”[6], 可以实现结果预测误差的降低。

同时, 本题模型建立时直接把历史的订单数可能出现的状态, 对未来一周的订单预测不一定从曾经出现过的订单状态中选出; 可以在计算马尔科夫状态概率时, 将各个状态拓展为状态尺度附件的范围, 再重复马尔科夫预测。

# 参考文献

[1] 周浩, 胡凌, 陈竹书. 基于量子遗传投影寻踪的长株潭地区耕地系统安全评价 [J/OL] [J]. 农业机械学报, 2022-06-23: 1-11.

[2] DAVIS L. Handbook of genetic algorithms[M]. CumInCAD, 1991: 2-8.

[3] 宋业新, 姜礼平, 陈绵云. 一类模糊线性规划模型的模糊最优区间值[J]. 模糊系统与数学, 2002, 16(2): 86-91.

[4] 张敏洪. 线性规划新算法的改进[J]. 高校应用数学学报: A 辑, 2000, 15(1): 101-106.

[5] 傅德胜, 张蓉. 一种改进的量子遗传算法研究[J]. 计算机仿真, 2013, 30(12): 321-325.

[6] 何成刚. 马尔科夫模型预测方法的研究及其应用[Z]. 合肥: 安徽大学, 2011.

# 附录 A 文件列表

表 11 **支撑材料**(上)

| 文件名 | 文件夹内容 | 文件描述 |
|---|---|---|
| answers.xlsx | | 四个问题的最终结果表格 (按工作表分开) |
| fitness.png | | 问题三中 DCQGA 算法适应度曲线 |
| 第一问 | | 问题一的程序包 |
| | code.m | 问题一的主程序 |
| | zb.m | 问题一求生产准备费用函数 |
| | storage money.mat | 问题一的库存费用数据 |
| | time.mat | 问题一的工时限制数据 |
| | need.mat | 问题一的订单需求数据 |
| | production money.mat | 问题一的生产准备费用数据 |
| 第二问 | | 问题二的程序包 |
| | code.m | 问题二的主程序 |
| | zb2.m | 问题二求生产准备费用函数 |
| | kc.m | 问题二求库存费用函数 |
| | storage money.mat | 问题一的库存费用数据 |
| | time.mat | 问题一的工时限制数据 |
| | need.mat | 问题一的订单需求数据 |
| | production money.mat | 问题一的生产准备费用数据 |

表 **12** **支撑材料(下)**

| 文件名 | 文件夹内容 | 文件描述 |
| --- | --- | --- |
| 第三问 | | 问题三的程序包 |
| | code.m | 问题三的主程序 |
| | bin2decfunction.m | 量子比特编码转十进制编码 |
| | kc.m | 问题三求库存费用函数 |
| | zb2.m | 问题三求生产准备费用函数 |
| | coll.m | 问题三中的二进制编码函数 |
| | gate.m | 问题三中的量子旋转门 |
| | goalfitness.m | 问题三的适应度函数 |
| | solvefitness.m | 问题三的适应度处理函数 |
| | solvefunction.m | 问题三的进制整合函数 |
| | Pop.m | 问题三中初始化编码函数 |
| | data.mat | 附表 7 中历史 30 周订单数据 |
| | storage money.mat | 问题三的库存费用数据 |
| | time.mat | 问题三的工时限制数据 |
| | need.mat | 问题三的订单需求数据 |
| | production money.mat | 问题三的生产准备费用数据 |
| 第四问 | | 问题四的程序包 |
| | code.m | 问题四的主程序 |
| | zb2.m | 问题二求生产准备费用函数 |
| | kc.m | 问题二求库存费用函数 |
| | mekf.m | 问题四中马尔科夫预测函数 |
| | new.m | 问题四中模拟测试集函数 |
| | judge.m | 问题四中的结果判断函数 |
| | data.mat | 附表 7 中历史 30 周订单数据 |
| | storage money.mat | 问题四的库存费用数据 |
| | time.mat | 问题四的工时限制数据 |
| | need.mat | 问题四的订单需求数据 |
| | ku.mat | 问题四的订单需求数据 |
| | production money.mat | 问题四的生产准备费用数据 |

# 附录 B　问题结果的报表模板

**附表 1: 问题 1 的结果**

| 日期 | WPCR 组装数量 | A 组装数量 | B 组装数量 | C 组装数量 | 生产准备费用 | 库存费用 |
|---|---|---|---|---|---|---|
| 周一 | | | | | | |
| 周二 | | | | | | |
| 周三 | | | | | | |
| 周四 | | | | | | |
| 周五 | | | | | | |
| 周六 | | | | | | |
| 周日 | | | | | | |
| 总和 | | | | | 此处填总成本 | |

**附表 2: 问题 2 的结果**

| 日期 | WPCR 组装数量 | A 组装数量 | B 组装数量 | C 组装数量 | 生产准备费用 | 库存费用 |
|---|---|---|---|---|---|---|
| 周一 | | | | | | |
| 周二 | | | | | | |
| 周三 | | | | | | |
| 周四 | | | | | | |
| 周五 | | | | | | |
| 周六 | | | | | | |
| 周日 | | | | | | |
| 总和 | | | | | 此处填总成本 | |

**附表 3: 问题 3 的结果 (检修日期及总成本)**

| 第 1 次 | 第 2 次 | 第 3 次 | 第 4 次 | 第 5 次 | 第 6 次 | 第 7 次 | 总成本 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

**附表 4: 问题 4 结果数据**

| 日期 | WPCR 组装数量 | A 组装数量 | B 组装数量 | C 组装数量 | 生产准备费用 | 库存费用 |
|---|---|---|---|---|---|---|
| 周一 | | | | | | |
| 周二 | | | | | | |
| 周三 | | | | | | |
| 周四 | | | | | | |
| 周五 | | | | | | |
| 周六 | | | | | | |
| 周日 | | | | | | |
| 总和 | | | | | 此处填总成本 | |

**附表 5: 每天 WPCR 需求和关键设备工时限制**

| 天 | 周一 | 周二 | 周三 | 周四 | 周五 | 周六 | 周日 |
|---|---|---|---|---|---|---|---|
| WPCR 需求（个） | 39 | 36 | 38 | 40 | 37 | 33 | 40 |
| A、B、C 生产总工时限制（工时） | 4500 | 2500 | 2750 | 2100 | 2500 | 2750 | 1500 |

**附表 6: 每次生产准备费用和单件库存费用**

| 产品 | WPCR | A | A1 | A2 | A3 | B | B1 | B2 | C | C1 | C2 | C3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 生产准备费用 | 240 | 120 | 40 | 60 | 50 | 160 | 80 | 100 | 180 | 60 | 40 | 70 |
| 单件库存费用 | 5 | 2 | 5 | 3 | 6 | 1.5 | 4 | 5 | 1.7 | 3 | 2 | 3 |

**附表 7: 问题 4 结果数据**

| 天 | 周一 | 周二 | 周三 | 周四 | 周五 | 周六 | 周日 |
|---|---|---|---|---|---|---|---|
| 第 1 周 | 39 | 36 | 38 | 40 | 37 | 33 | 40 |
| 第 2 周 | 39 | 33 | 37 | 43 | 34 | 30 | 39 |
| 第 3 周 | 42 | 36 | 35 | 38 | 36 | 35 | 41 |
| 第 4 周 | 38 | 36 | 36 | 48 | 34 | 35 | 39 |
| 第 5 周 | 38 | 36 | 40 | 40 | 40 | 34 | 39 |
| 第 6 周 | 40 | 30 | 36 | 40 | 34 | 36 | 37 |
| 第 7 周 | 41 | 36 | 41 | 41 | 38 | 29 | 43 |
| 第 8 周 | 33 | 31 | 40 | 42 | 42 | 30 | 40 |
| 第 9 周 | 35 | 36 | 38 | 33 | 35 | 37 | 41 |
| 第 10 周 | 43 | 35 | 42 | 37 | 36 | 33 | 39 |
| 第 11 周 | 38 | 32 | 41 | 36 | 40 | 31 | 34 |
| 第 12 周 | 37 | 37 | 41 | 39 | 38 | 35 | 38 |
| 第 13 周 | 38 | 38 | 33 | 42 | 42 | 29 | 33 |
| 第 14 周 | 39 | 37 | 44 | 38 | 35 | 36 | 38 |
| 第 15 周 | 40 | 39 | 38 | 38 | 37 | 34 | 44 |
| 第 16 周 | 35 | 36 | 38 | 39 | 39 | 39 | 39 |
| 第 17 周 | 43 | 28 | 39 | 41 | 38 | 30 | 38 |
| 第 18 周 | 35 | 37 | 40 | 41 | 40 | 35 | 41 |
| 第 19 周 | 36 | 35 | 40 | 41 | 37 | 38 | 36 |
| 第 20 周 | 37 | 38 | 39 | 41 | 38 | 37 | 44 |
| 第 21 周 | 37 | 37 | 37 | 36 | 39 | 33 | 41 |
| 第 22 周 | 39 | 37 | 42 | 37 | 36 | 28 | 43 |
| 第 23 周 | 40 | 32 | 35 | 45 | 40 | 34 | 43 |
| 第 24 周 | 38 | 36 | 37 | 36 | 40 | 28 | 45 |
| 第 25 周 | 38 | 40 | 38 | 36 | 35 | 40 | 42 |
| 第 26 周 | 31 | 31 | 44 | 36 | 31 | 36 | 40 |
| 第 27 周 | 40 | 36 | 34 | 43 | 35 | 32 | 39 |
| 第 28 周 | 33 | 33 | 36 | 41 | 34 | 38 | 40 |
| 第 29 周 | 35 | 34 | 37 | 37 | 39 | 36 | 40 |
| 第 30 周 | 37 | 41 | 39 | 41 | 36 | 32 | 44 |

# 附录 C　问题一–matlab 源程序

code.m

```
%% 第一问代码
%线性规划模型
clc;clear
load time.mat
load need.mat
load storage_money.mat
load production_money.mat
inter=1:28;
m=[7 6 5 4 3 2 1];
c=zeros(28,1);
c(2:4:26)=m*2;
c(3:4:27)=m*1.5;
c(4:4:28)=m*1.7;
c(1:4:25)=m*5-m*3*2-m*4*1.5-m*5*1.7;
A=zeros(56,28);
a=[0 3 5 5];
for i=1:7
    A(i,(i-1)*4+1:i*4)=a;
end


for i=8:14
    for j=1:i-7
        A(i,(j-1)*4+1)=-1;
        A(i+7,(j-1)*4+2)=-1/3;
        A(i+7*2,(j-1)*4+3)=-1/4;
        A(i+7*3,j*4)=-1/5;
    end
end
for i=1:3
    for j=1:7
        for k=1:j
            A(35+(i-1)*7+j,(k-1)*4+1)=1;
            if i==1
                A(35+(i-1)*7+j,(k-1)*4+1+i)=-1/3;
            elseif i==2
                A(35+(i-1)*7+j,(k-1)*4+1+i)=-1/4;
            else
                 A(35+(i-1)*7+j,(k-1)*4+1+i)=-1/5;
            end
        end
    end
end
```

```
aeq=zeros(4,28);
for i=1:7
    aeq(1,(i-1)*4+1)=1;
    aeq(2,(i-1)*4+2)=1/3;
    aeq(3,(i-1)*4+3)=1/4;
    aeq(4,i*4)=1/5;
end
beq=ones(4,1)*sum(need);
b=zeros(56,1);
b(1:7)=time';
b(8:14)=-cumsum(need)';
b(15:21)=-cumsum(need)';
b(22:28)=-cumsum(need)';
b(29:35)=-cumsum(need)';
vlb=zeros(1,28);
vub=ones(1,28)*inf;
x=intlinprog(c,inter,A,b,aeq,beq,vlb,vub);
out=zeros(7,4);
for i=1:7
    for j=1:4
        out(i,j)=x((i-1)*4+j);
    end
end
disp(out);
out_zb=zb(out,production_money);
```

zb.m

```
%% 此函数计算第一问的生产准备费用
function out=zb(x,production_money)
[m,n]=size(x);
out=zeros(m,1);
for i=1:m
   out(i)=x(i,1)*production_money(1)+x(i,2)*production_money(2)+x(i,3)*production_money(6)+x(i,4)*production_money(9);
   out(i)=out(i)+x(i,2)*(6*production_money(3)+8*production_money(4)+2*production_money(5))+x(i,3)*(2*production_money(7)+4*production_money(8))+x(i,4)*(8*production_money(10)+2*production_money(11)+12*production_money(12));
end
end
```

# 附录 D 问题二–matlab 源程序

code.

```
%% 第二问代码
%线性规划模型
clc;clear
load time.mat
load need.mat
load storage_money.mat
load production_money.mat
inter=1:28;
m=[7 6 5 4 3 2 1];
c=zeros(28,1);
c(2:4:26)=m*2+66;
c(3:4:27)=m*1.5+28;
c(4:4:28)=m*1.7+64;
c(1:4:25)=m*5-m*3*2-m*4*1.5-m*5*1.7;
c(1)=c(1)+7*(3*2+4*1.5+5*1.7);
A=zeros(56,28);
a=[0 3 5 5];
for i=1:7
    A(i,(i-1)*4+1:i*4)=a;
end


for i=8:14
    for j=1:i-7
        A(i,(j-1)*4+1)=-1;
        A(i+7,(j-1)*4+2)=-1/3;
        A(i+7*2,(j-1)*4+3)=-1/4;
        A(i+7*3,j*4)=-1/5;
    end
end
for i=1:3
    for t=1:7
        A(35+(i-1)*7+1,(t-1)*4+1)=1;
         if i==1
                A(35+(i-1)*7+1,(t-1)*4+1+i)=-1/3;
            elseif i==2
                A(35+(i-1)*7+1,(t-1)*4+1+i)=-1/4;
            else
                 A(35+(i-1)*7+1,(t-1)*4+1+i)=-1/5;
         end
    end
    for j=2:7
        for k=2:j
            A(35+(i-1)*7+j,(k-1)*4+1)=1;
            if i==1
                A(35+(i-1)*7+j,mod(k+5,7)*4+1+i)=-1/3;
            elseif i==2
```

```
                A(35+(i-1)*7+j,mod(k+5,7)*4+1+i)=-1/4;
            else
                A(35+(i-1)*7+j,mod(k+5,7)*4+1+i)=-1/5;
            end
        end
    end
end

aeq=zeros(4,28);
for i=1:7
    aeq(1,(i-1)*4+1)=1;
    aeq(2,(i-1)*4+2)=1/3;
    aeq(3,(i-1)*4+3)=1/4;
    aeq(4,i*4)=1/5;
end
beq=ones(4,1)*sum(need);
b=zeros(56,1);
b(1:7)=time';
b(8:14)=-cumsum(need)';
b(15:21)=-cumsum(need)';
b(22:28)=-cumsum(need)';
b(29:35)=-cumsum(need)';
vlb=zeros(1,28);
vub=ones(1,28)*inf;
x=intlinprog(c,inter,A,b,aeq,beq,vlb,vub);
out=zeros(7,4);
for i=1:7
    for j=1:4
        out(i,j)=x((i-1)*4+j);
    end
end
disp(out);
out_zb2=zb_2(out,production_money);
out_kc=kc(out,storage_money,need);
```

zb2.m

```
%% 此函数计算第二问的生产准备费用
function out=zb_2(x,production_money)
[m,n]=size(x);
out=zeros(m,1);
for i=1:m
   out(i)=x(i,1)*production_money(1)+x(i,2)*production_money(2)+x(i,3)*production_money(6)+x(i,4)*production_money(9);
   out(i)=out(i)+x(mod(i,m)+1,2)*(6*production_money(3)+8*production_money(4)+2*production_money(5))+x(mod(i,m)+1,3)*(2*production_money(7)+4*production_money(8))+x(mod(i,m)+1,4)*(8*production_money(10)+2*production_money(11)+12*production_money(12));
```

```
end
end
```

kc.m

```
%% 此函数用于计算第二问的库存费用
function out=kc(x,storage_money,need)
[m,n]=size(x);
out=zeros(m,1);
t=[0 117 156 195];
for i=1:m
   t(2:end)=t(2:end)+x(i,2:end)-x(i,1)*[3 4 5];
   t(1)=t(1)+x(i,1)-need(i);
   out(i)=t(1)*storage_money(1)+t(2)*storage_money(2)+t(3)*storage_money(6)+t(4)*
    storage_money(9);
   out(i)=out(i)+x(mod(i,m)+1,2)*(6*storage_money(3)+8*storage_money(4)+2*
    storage_money(5))+x(mod(i,m)+1,3)*(2*storage_money(7)+4*storage_money(8))+x(mod(
    i,m)+1,4)*(8*storage_money(10)+2*storage_money(11)+12*storage_money(12));
end
end
```

# 附录 E　问题三–matlab 源代码

code.m

```
%% 第三问代码
%量子遗传优化算法(QGA)
clc;clear
tic
%% 参数设置
ger=100;%遗传的代数目
n=20;%种群数目
n_chr=7;%染色体数目
global range
range=ones(7,2).*[1 210];   % 函数自变量的范围既染色体取值范围
range_len=ones(1,7)*8;%每个二进制编码的长度
ger_fitness=zeros(1,ger);%每一代的最佳适应度
best=struct('fitness',0,'dec',[],'bin',[],'chr',[]);%记录最佳个体的适应度值、十进制
    值、二进制编码、量子比特编码
%          适应度      十进制值   二进制码     量子比特编码
%初始化种群
chr=Pop(n*2,sum(range_len));
%得到二进制编码
bin=coll(chr);
%计算初始种群的适应度和十进制编码
[fitness,dec]=solvefitness(bin,range_len);
%% 记录最佳个体到best
```

```
[best.fitness,bestid]=min(fitness);     % 找出最大值
best.bin=bin(bestid,:);
best.chr=chr(2*bestid-1:2*bestid,:);
best.dec=dec(bestid,:);
ger_fitness(1)=best.fitness;
fprintf('第%d代遗传开始\n',1)

%% 开始遗传
for i=2:ger
    fprintf('第%d代遗传开始\n',i);
    bin=coll(chr);
    [fitness,dec]=solvefitness(bin,range_len);
    chr=gate(chr,fitness,best,bin);%量子旋转门
    [new_fitness,newfitness_id]=min(fitness);
    %更新最优适应度下的最优个体
    if new_fitness<best.fitness
        best.fitness=new_fitness;
        best.bin=bin(newfitness_id,:);
        best.chr=chr(2*newfitness_id-1:2*newfitness_id,:);
        best.dec=dec(newfitness_id,:);
    end
    ger_fitness(i)=best.fitness;
end
disp('遗传结束')
%% 画进化曲线
figure(1)
plot(1:ger,ger_fitness,'b-','LineWidth',1.5)
title('量子遗传优化算法');
xlabel('进化代数');
ylabel('每代的最佳适应度');

%% 显示优化结果
disp(['最优解： ',num2str(best.dec)])
disp(['最佳适应度:',num2str(best.fitness)]);
toc
```

zb2.m

```
%% 此函数计算第二问的生产准备费用
function out=zb_2(x,production_money)
[m,n]=size(x);
out=zeros(m,1);
for i=1:m
   out(i)=x(i,1)*production_money(1)+x(i,2)*production_money(2)+x(i,3)*
    production_money(6)+x(i,4)*production_money(9);
   out(i)=out(i)+x(mod(i,m)+1,2)*(6*production_money(3)+8*production_money(4)+2*
    production_money(5))+x(mod(i,m)+1,3)*(2*production_money(7)+4*production_money
    (8))+x(mod(i,m)+1,4)*(8*production_money(10)+2*production_money(11)+12*
```

```
    production_money(12));
end
end
```

solvefunction.m

```
function [fitness,dec]=solvefunction(bin,range_len)
%% 目标函数
% 输入      bin: 二进制编码
%   range_len: 各变量的二进制位数
% 输出      fitness: 目标值
%           dec: 十进制数
global range;
%% 二进制转十进制
dec=bin2decfunction(bin,range_len,range);
%% 计算适应度-函数值
fitness=goal_fitness(dec);
```

solvefitness.m

```
%% 适应度函数
% 输入   bin: 二进制编码
%      range_len: 各变量的二进制位数
% 输出 fitness: 适应度
%             dec: 十进制数（待优化参数）
function [fitness,dec]=solvefitness(bin,range_len)
sizepop=size(bin,1);
fitness=zeros(1,sizepop);
num=size(range_len,2);
dec=zeros(sizepop,num);
for i=1:sizepop
    [fitness(i),dec(i,:)]=solvefunction(bin(i,:),range_len); %solvefunction题目所设
    目标函数
end
```

Pop.m

```
%量子比特编码
function chr=Pop(m,n)
for i=1:m
    for j=1:n
        chr(i,j)=1/sqrt(2);
    end
end
end
```

kc.m

```
%% 此函数用于计算第二问的库存费用
function out=kc(x,storage_money,need)
[m,n]=size(x);
```

```
out=zeros(m,1);
t=[0 117 156 195];
for i=1:m
   t(2:end)=t(2:end)+x(i,2:end)-x(i,1)*[3 4 5];
   t(1)=t(1)+x(i,1)-need(i);
   out(i)=t(1)*storage_money(1)+t(2)*storage_money(2)+t(3)*storage_money(6)+t(4)*
    storage_money(9);
   out(i)=out(i)+x(mod(i,m)+1,2)*(6*storage_money(3)+8*storage_money(4)+2*
    storage_money(5))+x(mod(i,m)+1,3)*(2*storage_money(7)+4*storage_money(8))+x(mod(
    i,m)+1,4)*(8*storage_money(10)+2*storage_money(11)+12*storage_money(12));
end
end
```

goalfitness.m

```
%% 适应度函数
function fitness=goal_fitness(dec)
load data.mat;
load need.mat;
load time.mat;
load storage_money.mat;
load production_money.mat;
new_data=zeros(210,1);
new_time=zeros(210,1);
for i=1:30
    for j=1:7
        new_data((i-1)*7+j)=data(i,j);
        new_time((i-1)*7+j)=time(j);
    end
end
a=10:-2:0;
for i=1:7
    new_time(dec(i))=0;
    for j=1:6
        if dec(i)+j<=210
        new_time(dec(i)+j)=new_time(dec(i)+j)*(1+0.01*a(j));
        end
    end
end
m=210:-1:1;
inter=1:840;
c=zeros(840,1);
c(2:4:end)=m*2+66;
c(3:4:end)=m*1.5+28;
c(4:4:end)=m*1.7+64;
c(1:4:end)=m*5-m*3*2-m*4*1.5-m*5*1.7;
c(1)=c(1)+210*(3*2+4*1.5+5*1.7);
A=zeros(1677,840);
```

```
a=[0 3 5 5];
for i=1:210
    A(i,(i-1)*4+1:i*4)=a;
end
for i=210+1:420
    for j=1:i-210
        A(i,(j-1)*4+1)=-1;
        A(i+210,(j-1)*4+2)=-1/3;
        A(i+210*2,(j-1)*4+3)=-1/4;
        A(i+210*3,j*4)=-1/5;
    end
end
for i=1:3
   for j=1:209
       for k=1:j
           A(1050+(i-1)*209+j,k*4+1)=1;
            if i==1
                A(1050+(i-1)*209+j,(k-1)*4+2)=-1/3;
            elseif i==2
                A(1050+(i-1)*209+j,(k-1)*4+3)=-1/4;
            elseif i==3
                 A(1050+(i-1)*209+j,(k-1)*4+4)=-1/5;
            end
       end
   end
end
b=zeros(1677,1);
b(1:210)=new_time';
for i=1:4
    b(i*210+1:(i+1)*210)=-cumsum(new_data);
end
aeq=zeros(8,840);
for i=1:7
    aeq(i+1,(dec(i)-1)*4+1)=1;
end
for i=1:210
    aeq(1,(i-1)*4+1)=1;
end
beq=zeros(8,1);
beq(1)=sum(new_data);
vlb=zeros(1,840);
options = optimoptions('intlinprog','MaxFeasiblePoints',1,'Heuristics','
    intermediate');
x=intlinprog(c,inter,A,b,aeq,beq,vlb,[],[],options);
if isempty(x)==0
out=zeros(210,4);
for i=1:210
```

```
    for j=1:4
        out(i,j)=x((i-1)*4+j);
    end
end
out_zb2=zb_2(out,production_money);
out_kc=kc(out,storage_money,new_data);
fitness=sum(out_zb2)+sum(out_kc);
else
    fitness=inf;
end
end
```

gate.m

```
%% 量子旋转门调整策略
% 输入  chr: 更新前的量子比特编码
%     fitness: 适应度值
%        best: 当前种群中最优个体
%       bin: 二进制编码
% 输出  chr: 更新后的量子比特编码
function chr=gate(chr,fitness,best,bin)
size_pop=size(chr,1)/2;
len_chr=size(bin,2);
for i=1:size_pop
    for j=1:len_chr
        alpha=chr(2*i-1,j);   % α自旋向上
        beta=chr(2*i,j);     % β自旋向上
        x=bin(i,j);
        b=best.bin(j);
        if ((x==0)&&(b==0))||((x==1)&&(b==1))
            delta=0;                    % delta为旋转角的大小
            s=0;                          % s为旋转角的符号, 即旋转方向
        elseif (x==0)&&(b==1)&&(fitness(i)<best.fitness)
            delta=0.01*pi;
            if alpha*beta>0
                s=1;
            elseif alpha*beta<0
                s=-1;
            elseif alpha==0
                s=0;
            elseif beta==0
                s=sign(randn);
            end
        elseif (x==0)&&(b==1)&&(fitness(i)>=best.fitness)
            delta=0.01*pi;
            if alpha*beta>0
                s=-1;
            elseif alpha*beta<0
```

```
                s=1;
            elseif alpha==0
                s=sign(randn);
            elseif beta==0
                s=0;
            end
        elseif (x==1)&&(b==0)&&(fitness(i)<best.fitness)
            delta=0.01*pi;
            if alpha*beta>0
                s=-1;
            elseif alpha*beta<0
                s=1;
            elseif alpha==0
                s=sign(randn);
            elseif beta==0
                s=0;
            end
        elseif (x==1)&&(b==0)&&(fitness(i)>=best.fitness)
            delta=0.01*pi;
            if alpha*beta>0
                s=1;
            elseif alpha*beta<0
                s=-1;
            elseif alpha==0
                s=0;
            elseif beta==0
                s=sign(randn);
            end
        end
        e=s*delta;       % e为旋转角
        U=[cos(e) -sin(e);sin(e) cos(e)];      % 量子旋转门
        y=U*[alpha beta]';        % y为更新后的量子位
        chr(2*i-1,j)=y(1);
        chr(2*i,j)=y(2);
    end
end
```

coll.m

```
%得到二进制编码函数
function bin=coll(chr)
[m,n]=size(chr);
m=m/2;%得到种群大小
bin=zeros(m,n);%二进制编码初始化
for i=1:m
    for j=1:n
        r=rand;
        if r>(chr(2.*i-1,j)^2)  %随机数大于α平方（自旋向上）
```

```
            bin(i,j)=1;
        else
            bin(i,j)=0;
        end
    end
end
end
```

bin2decfunction.m

```
%% 二进制转十进制
% 输入      bin: 二进制编码
%     range_len: 各变量的二进制位数
%         range: 各变量的范围
% 输出      dec: 十进制数
function dec=bin2decfunction(bin,range_len,range)

m=length(range_len);
n=1;
dec=zeros(1,m);
for i=1:m
    for j=range_len(i)-1:-1:0
        dec(i)=dec(i)+bin(n).*2.^j;
        n=n+1;
    end
end
dec=range(:,1)'+dec./(2.^range_len-1).*(range(:,2)-range(:,1))';
dec=sort(dec);
for i=1:7
    dec(i)=floor(dec(i));
end
for i=1:6
    if dec(i+1)-dec(i)<6
        dec(i+1)=dec(i)+6;
    end
    if dec(i+1)>210
        dec(i+1)=210;
    end
end
end
```

# 附录 F 问题四–matlab 源代码

code.m

```

%% 第四问代码
```

```
%线性规划模型
clc;clear
load time.mat
load storage_money.mat
load production_money.mat
load data.mat
new_data = new(data,100000);
out=zeros(700000,4);
for l=1:100000
    need=new_data(l,:);
inter=1:28;
m=[7 6 5 4 3 2 1];
c=zeros(28,1);
c(2:4:26)=m*2+66;
c(3:4:27)=m*1.5+28;
c(4:4:28)=m*1.7+64;
c(1:4:25)=m*5-m*3*2-m*4*1.5-m*5*1.7;
c(1)=c(1)+7*(3*2+4*1.5+5*1.7);
A=zeros(56,28);
a=[0 3 5 5];
for i=1:7
    A(i,(i-1)*4+1:i*4)=a;
end


for i=8:14
    for j=1:i-7
        A(i,(j-1)*4+1)=-1;
        A(i+7,(j-1)*4+2)=-1/3;
        A(i+7*2,(j-1)*4+3)=-1/4;
        A(i+7*3,j*4)=-1/5;
    end
end
for i=1:3
    for t=1:7
        A(35+(i-1)*7+1,(t-1)*4+1)=1;
         if i==1
                A(35+(i-1)*7+1,(t-1)*4+1+i)=-1/3;
            elseif i==2
                A(35+(i-1)*7+1,(t-1)*4+1+i)=-1/4;
            else
                 A(35+(i-1)*7+1,(t-1)*4+1+i)=-1/5;
         end
    end
    for j=2:7
        for k=2:j
            A(35+(i-1)*7+j,(k-1)*4+1)=1;
```

```
            if i==1
                A(35+(i-1)*7+j,mod(k+5,7)*4+1+i)=-1/3;
            elseif i==2
                A(35+(i-1)*7+j,mod(k+5,7)*4+1+i)=-1/4;
            else
                 A(35+(i-1)*7+j,mod(k+5,7)*4+1+i)=-1/5;
            end
        end
    end
end

aeq=zeros(4,28);
for i=1:7
    aeq(1,(i-1)*4+1)=1;
    aeq(2,(i-1)*4+2)=1/3;
    aeq(3,(i-1)*4+3)=1/4;
    aeq(4,i*4)=1/5;
end
beq=ones(4,1)*sum(need);
b=zeros(56,1);
b(1:7)=time';
b(8:14)=-cumsum(need)';
b(15:21)=-cumsum(need)';
b(22:28)=-cumsum(need)';
b(29:35)=-cumsum(need)';
vlb=zeros(1,28);
vub=ones(1,28)*inf;
x=intlinprog(c,inter,A,b,aeq,beq,vlb,vub);
for i=1:7
    for j=1:4
        out((l-1)*7+i,j)=x((i-1)*4+j);
    end
end
end
train=zeros(7,4);
flag_i=ones(100000,1);
best_fitness=inf;
best=zeros(7,4);
best_i=0;
for i=1:100000
    for j=1:7
        train(j,:)=out((i-1)*7+j,:);
    end
    [judge_1,judge_2]=judge(train,new_data);
    if judge_1==0||judge_2==0
        flag_i(i)=0;
    end
```

```
end
for i=1:100000
    if flag_i(i)==1
    for j=1:7
        train(j,:)=out((i-1)*7+j,:);
    end
    fitness=sum(kc(train,storage_money,new_data(i,:),train(1,1)*[0 3 4 5]))+sum(
    zb_2(train,production_money));
    if fitness<best_fitness
        best_fitness=fitness;
        best=train;
        best_i=i;
    end
    end
end
best_zb=zb_2(best,production_money);
best_kc=kc(best,storage_money,new_data(best_i,:),best(1,1)*[0 3 4 5]);
disp('生产准备费用')
disp(best_zb)
disp('库存费用')
disp(best_kc)
```

judge.m

```
%% 此函数用于判断日和周的概率能否满足题设要求
function [judge_1,judge_2]=judge(train,new_data)
count1=0;
count2=0;
for i=1:100000
    t=0;
    flag=1;
    need=new_data(i,:);
    for j=1:7
        t=t+train(j,1);
        if t-need(j)>=0
            t=t-need(j);
            count1=count1+1;
        else
            t=0;
            flag=0;
        end
    end
        if flag==1
            count2=count2+1;
        end
end
if count1/700000>=0.95
    judge_1=1;
```

```
else
    judge_1=0;
end
if count2/100000>=0.85
    judge_2=1;
else
    judge_2=0;
end
end
```

kc.m

```
%% 此函数用于计算第二问的库存费用
function out=kc(x,storage_money,need,t)
[m,n]=size(x);
out=zeros(m,1);

for i=1:m
   t(2:end)=t(2:end)+x(i,2:end)-x(i,1)*[3 4 5];
   t(1)=t(1)+x(i,1)-need(i);
   out(i)=t(1)*storage_money(1)+t(2)*storage_money(2)+t(3)*storage_money(6)+t(4)*
    storage_money(9);
   out(i)=out(i)+x(mod(i,m)+1,2)*(6*storage_money(3)+8*storage_money(4)+2*
    storage_money(5))+x(mod(i,m)+1,3)*(2*storage_money(7)+4*storage_money(8))+x(mod(
    i,m)+1,4)*(8*storage_money(10)+2*storage_money(11)+12*storage_money(12));
end
end
```

mekf.m

```
%% 马尔可夫预测
function [out,n]=mekf(k,data)
    test_data=zeros(30,2);
    test_data(:,1)=data(:,end);
    test_data(:,2)=data(:,k);
    c=unique(test_data(:,2));
   n=numel(c);%状态类型个数

   y=zeros(30,1);
   t=0;
   out=zeros(n,2);
for ii=1:30
[y(ii),t]=find(c==test_data(ii,2));
end
x=test_data(:,1);
num=length(y);%得到状态总个数
%状态转移概率矩阵
p=zeros(n,n);%初始化元素分子
c_num=zeros(n,1);%状态起始数量
```

```
for i=1:29
     c_num(y(i))=c_num(y(i))+1;
     p(y(i),y(i+1))=p(y(i),y(i+1))+1;

end
for i=1:n
    p(i,:)=p(i,:)/c_num(i);
end
for i=1:n
    if isnan(p(i,1))
        p(i,:)=1/n;
    end
end
% 马尔可夫预测

predict=1;%需要预测的期数
pi=zeros(predict+1,n);%初始化状态概率
 %得到起始状态概率
   for i=1:n
       if y(end)==i
           pi(1,i)=1;
       end
   end
   %生成后续每期的状态概率
   for i=2:predict+1
       pi(i,:)=pi(i-1,:)*p;
   end
  %% 终极概率预测
   result = null(eye(n)-p');
   result = result/sum(result);
   out(:,1)=c;
   out(:,2)=result;
%    disp(out)
end
```

new.m

```
%% 蒙特卡洛模拟
function new_data=new(data,nn)
new_data = zeros(nn,7);
% 模拟
for l=1:nn
for i=1:7
    a=rand;
    [out,n]=mekf(i,data);
    t=cumsum(out(:,2));
    for j=1:n
if a<=t(j)
```

```
    new_data(l,i)=out(j,1);
    break;
end
    end
end
end
```

zb2.m

```
%% 此函数计算第二问的生产准备费用
function out=zb_2(x,production_money)
[m,n]=size(x);
out=zeros(m,1);
for i=1:m
   out(i)=x(i,1)*production_money(1)+x(i,2)*production_money(2)+x(i,3)*
    production_money(6)+x(i,4)*production_money(9);
   out(i)=out(i)+x(mod(i,m)+1,2)*(6*production_money(3)+8*production_money(4)+2*
    production_money(5))+x(mod(i,m)+1,3)*(2*production_money(7)+4*production_money
    (8))+x(mod(i,m)+1,4)*(8*production_money(10)+2*production_money(11)+12*
    production_money(12));
end
end
```