队伍编号	MC2309998
题号	C

基于外层遗传内层模拟退火的电商物流网络调度优化方法

摘 要

当今时代，随着信息科技的飞速发展，电商物流行业方兴未艾，由于疫情、地震和各种促销节的影响，物流网络的运货量极不稳定，难以保证物流网络在未来不会发生部分瘫痪。因此根据已有历史数据来预测物流量的未来走向和确立场地临时停运的紧急调度方案尤其重要。

问题 1 需要本文根据两年数据来预测未来一个月的所有线路的运货量。为此本文首先对所有线路的基本信息进行统计（包括最大流量，流量存在天数，流量变化等），发现大量线路的运货量信息都会受异常值影响。为了消除特殊因素的影响，本文建立了特定的**数据预处理**方法。处理之后对每条线路利用***ARMA*预测**方法进行了预测。将两年数据中最近十天的数据用于验证，平均误差率约 10%。最后得出了所有线路的预测结果，并列举了$DC14 \to DC10$、$DC20 \to DC35$、$DC25 \to DC62$三条边的预测结果。

问题 2 要求在问题 1 预测结果的基础上，给出因$DC5$停运而实施的调度方案。在一些假设下，为了方便模型的描述，本文首先定义几类与$DC5$有关场地和线路的集合与变量。其次构建了是否存在满足运货量的可行方案的判断准则，然后给出了目标函数和约束条件的表达式，最后针对$DC5$场地的收货量和发货量分别建立了**双目标优化约束模型**，再化简为**等价的单目标优化模型**。采用**外层遗传算法、内层模拟退火算法**的方式计算求得最优分流方案。

问题 3 要求在问题 1 预测结果的基础上，给出因$DC9$停运而采取的方案。与问题 2 不同的是，问题 3 允许新开或关闭线路。***DC*9具有特殊性**，其在 2021 年的流量较大，在 2022 年的流量较小接近于 0 但在近期有恢复的趋势。因此本题模型不仅分配$DC9$恢复的流量，在**分配**流量的基础上对$DC9$附近的网络流进行了**优化**。本问题首先构建了是否存在满足运货量的可行方案的判断准则，然后建立优化模型并将其化简为**单目标优化模型。**最后采用和问题 2 一致的算法求解最优解。

问题 4 首先要求对网络场地和运输线路进行评价，再给出新建设场地和建设运输线路的最优方案并判断所建网络的鲁棒性。本问核心在于不仅需要确定建设场地和线路的数量，还需确定各条新增线路的运货量和新增场地的处理能力。为此本文首先针对场地和运输线路选取了 12 个、8 个指标，建立基于**主成分分析**的**重要性评价模型**得到原有场地处理能力和线路运输能力的评价函数和评价结果。根据实际情况计算建设成本，并使用遗传和模拟退火相结合的算法得到网络流的**最优调度**。在合理的假设下，得到该调度的评价函数，进而给出建设成本最小且建设效益最大的目标函数。最后分析该网络的鲁棒性。

本文所构建的模型优点在于合理的数据预处理、合理简化问题且构建的模型详细全面等。最后对问题 4 的方案进行鲁棒性分析发现，当目标函数下降 20%以内被接受时，允许网络中货量扰动不超过 130%且求得的结果中新增线路的变化情况也很少，说明网络的**鲁棒性较强**。

关键词：ARMA 预测；双目标优化；遗传算法；模拟退火；主成分分析

目录

一、问题重述

1.1 问题背景

21 世纪以来，电子商务随着经济的高速发展而逐渐兴起。随着物品需求量的增多，电子商务的物流网络也得到了进一步的构建。电子商务的物流网络主要由包含接货仓、分拣中心、营业部在内的物流场地和它们之间的连接线路组成，其中连接线路一般被称为运输线路。运输过程中的货量受多方面因素的影响，当有节假日或促销活动时，货量受电子商务客户群体的下单数量影响相对变多；当受疫情等天灾人祸影响时，货量因物流场地停用而分配到其他地点，简介影响了其他运输线路的运输和物流场地的处理。

虽然电子商务的物流网络日益完善，但还存在着一些问题。为提高运输效率、降低运营成本，需要对物流场地和运输线路的货量进行预测，提前进行安排。为降低不可抗因素导致物流网络某些环节无法使用所造成的影响，当某些场地因为不可抗因素暂时或永远无法使用时，还要以预测结果和各场地及运输线路的处理及运输能力为基础做出相应的调整。因此如果能准确预测货量、精确衡量物流场地的处理能力和运输线路的运输能力，并建立相应的模型安排日常情况、给出调整突发情况的运输方案，将大幅提升电子商务物流网络的发展。

1.2 问题提出

围绕电子商务物流网络的日常结构安排和突发情况处理进行分析并建立数学模型，本文依次解决如下问题：

问题 1：基于本题中给出的 2021 年 1 月 1 日到 2022 年 12 月 31 日的 81 个物流场地和 1049 条有向运输线路所对应的货量数据，预测 2023 年 1 月 1 日到 2023 年 1 月 31 日过程中每天每条线路的货量，并给出以下 3 条运输线路的预测结果：$DC14 \rightarrow DC10$、$DC20 \rightarrow DC35$、$DC25 \rightarrow DC62$。

问题 2：在 2023 年 1 月 1 日关闭$DC5$，以问题 1 预测的结果为基础建立模型提出分配方案，将与该物流场地相联系的运输线路的货物分配给其他线路。要求满足：①尽量正常流转，即没有货物滞留；②关闭$DC5$后货量发生改变的运输线路尽量少；③保持每条运输线路的工作负荷尽量平衡，即改变线路负荷尽可能相同。如果能够正常流转，给出因关闭$DC5$导致货量变化的运输线路条数和网络的负荷情况；如果存在一些货物不能正常流转，要在满足上述①到③的要求下，满足 2023 年 1 月 1 日到 2023 年 1 月 31 日之间的不能正常流转的货物每天的累计总量尽量少，并给出发生变化的运输线路条数、不能正常流转的货量和网络的负荷情况。

问题 3：在问题 2 的基础上，将关闭$DC5$更改为关闭$DC9$，允许对物流网络每天进行调整，方式为关闭或新开线路但不能新增物流场地，并将与$DC9$相关运输线路的货物

分配给其他线路。其中新开的运输线路运输能力的上限为当前有的运输线路的最大值。在正常流转和不能正常流转的情况下均满足问题 2 所述的要求，在正常流转的情况下，给发货量变化的运输线路条数和网络的负荷情况；在不能正常流转的情况下，给出发生变化的运输线路条数、不能正常流转的货量和网络的负荷情况。

问题 4：根据本题中给出的初始数据，对物流场地和运输线路进行重要性评价。结合问题 1 的结果新增物流场地与线路，给出新增的场地和线路以及他们的处理能力和运输能力，并给出问题 4 所建立的物流网络的鲁棒性。

其中带箭头的表示运输线路，DC开头的表示物流场地。每个物流场地的处理能力和每条线路的运输能力上限均为其历史货量的最大值。

二、模型假设

假设一：2023-01-01 至 2023-01-31 期间不会发生特殊事件，如疫情地震等。

假设二：每个场地的处理能力定义为最大货物接收量和最大货物发出量，上限由历史最大值决定。

假设三：新增场地为一个，若新增多个场地，方式为建设完一个后再建设下一个。

假设四：建设运输线路的成本只与新增线路的数量有关。

假设五：新增线路的工作负荷量为现有线路负荷量的平均值。

假设六：新增线路每天的运货量均为定值。

假设七：新增物流场地的最大收货能力为所有新增线路通向该场地的运输线路的最大运货量之和；新增场地的最大发货能力为所有新增线路中从该场地出发的线路最大运货量之和。

假设八：与新建发货线路连接的场地当天总收获量不变，此场地其他已有收货线路的运货量减小相同的百分比，使得减小量等于新增运货线路的运货量；与新建收货线路连接的场地当天总发获量不变，此场地其他已有发货线路的运货量减小相同的百分比，使得减小量等于新增运货线路的运货量。

三、符号说明

符号	说明
DCi	第i个场地
$DCi \rightarrow DCj$	第i个场地到第j个场地的运输线路
$d_{i,j}$	可达值，判断是否存在运输线路
$x_{i,j,t}$	第t天从物流场地i到物流场地j的货量

Q_1	下四分位数
Q_2	上四分位数
IQR	四分位数
q_s	$Spearman$相关系数
C	建设成本
A	建设效益

四、问题 1 的模型建立求解

4.1 问题 1 的分析

问题 1 是一道预测题，本题需要根据每条运输线路 2021 年 1 月 1 日到 2022 年 12 月 31 日期间 730 天的数据来预测 2023 年 1 月 1 日到 2023 年 1 月 31 日期间 31 天的数据，考虑到一些突发因素会对物流量的大小产生极大的影响，对有向运输线路的 730 组数据进行数据预处理，在模型的准备阶段对运输线路的可达性进行筛选，筛去零值过多和最近未发货的运输线路，用剩余运输线路的货量数据作为原始数据进行训练。同时考虑到这些数据代表的某一天的物流量与相邻天的物流量具有关联性。为此，我们可以先绘制箱线图对每一条运输线路的数据进行异常值的剔除，并把筛选出来的数据分为训练集和验证集。然后基于$ARMA$建立货量预测模型，并对训练集进行预测，再利用验证集的数据进行调参，最后利用模型进行预测。

问题 1 流程图如图 4.1 所示。

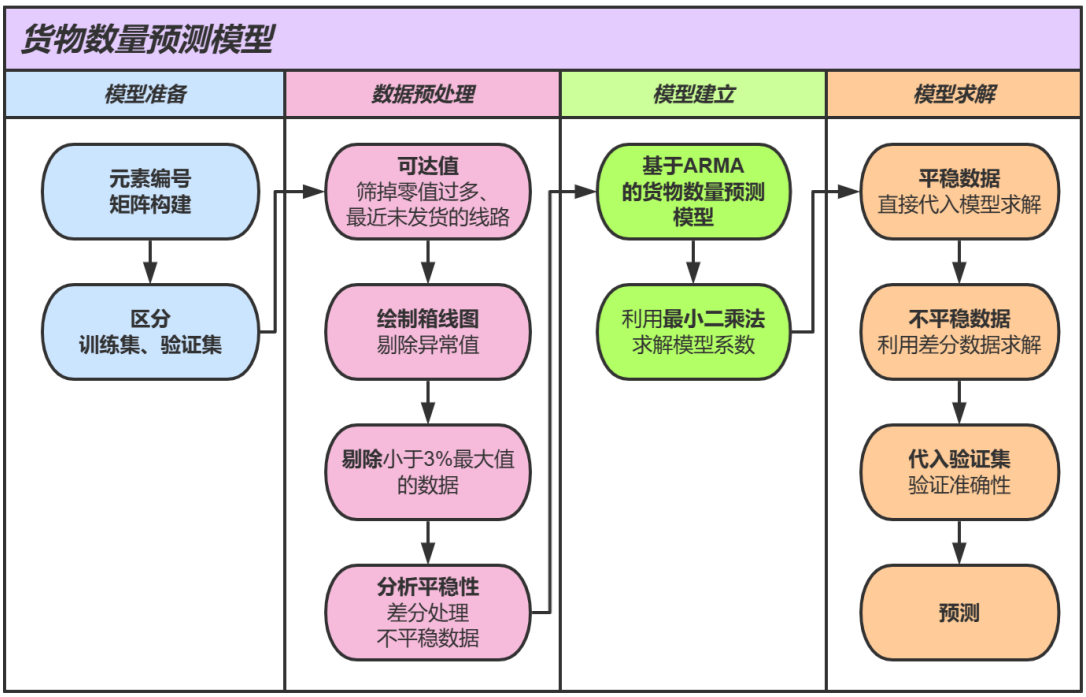

图 4.1　问题 1 流程图

4.2 模型准备

4.2.1 元素编号及矩阵构建

观察赛题附件数据文件中场地 1、场地 2 的序号可知，$DC3$表示物流第 3 个场地，这种表示方法简单明了，但会对使用软件求解造成一定困难。因此在模型准备阶段，直接提取出物流场地DCi的编号i表示此物流场地。

进一步观察发现场地 1、场地 2 均未用到所有 81 个物流场地，计算物流场地间的连通性。因为线路是有向的，当物流场地DCi到物流场地DCj连通是记为 1，不连通记为 0，由此得到 81 个物流场地之间的连通矩阵。

由赛题数据附件数据文件“货量”列可知 1049 条运输线路每天的货量，为方便后续模型建立及求解，可以基于每天的数据可以构建运输线路货量矩阵。此部分代码在附件。

4.2.2 区分训练集、验证集

将 2021-01-01 至 2022-12-21 期间经过数据清洗的运输线路货量数据作为训练集，2021-01-01 至 2022-12-31 期间最后 10 天的数据作为验证集，用训练集进行训练，用验证集验证模型的准确性，考察模型泛化性。

4.3 数据预处理

4.3.1 筛掉零值过多和最近未发货的运输线路

记$x_{i,j,t}$为第t天从物流场地i到物流场地j的货量，其中$i,j = 1,2,\cdots,81$，$t = 1,2,\cdots 730$。

首先需要判断物流场地i到物流场地j是否存在运输线路，定义可达值$d_{i,j}$：

$$d_{i,j} = \begin{cases} 1, & \exists t, s.t.\ x_{i,j,t} \neq 0 \\ 0, & \forall t, s.t.\ x_{i,j,t} = 0 \end{cases} \tag{4.1}$$

当$d_{i,j} = 0$时，运输线路的流量恒为零，即不可达，因此在后续货量预测预测模型中只需要考虑$d_{i,j} \neq 0$所对应的运输线路即可。

其次，在$d_{i,j} \neq 0$所对应的运输线路中，经绘图发现存在以下两种情况的运输线路：

① 最近较多天数（> 100）内运货量为零的点；

② 730 天内只存在极少天数（< 5）存在运货量的点。

因为运输线路在这两种情况时，最终预测结果要么等于零要么极小接近于零，因此同样在问题 1 后续模型建立求解中同样不考虑这两种情况的运输线路。

4.3.2 箱线图剔除异常值

在本问题中，对每一天的运输线路进行时间序列预测，在模型准备中利用可达值筛掉流量恒为零的不可达线路后，对可达运输线路数据进行预处理，具体操作为绘制利用箱线图剔除异常值数据。

首先计算运输线路的上、下四分位数Q_2、Q_1和四分位距IQR：

Q_1：下四分位数，数据由小到大排列后第 25%位置的数据；

Q_2：上四分位数，数据由小到大排列后第 75%位置的数据；

IQR：四分位距，$IQR = Q_2 - Q_1$。

在箱线图中，将满足$x_{i,j,t} > Q_2 + 1.5IQR$或$x_{i,j,t} < Q_1 - 1.5IQR$的运输线路作为异常值并剔除，在后续模型求解中不予考虑。

4.3.3 剔除小于 3%最大值的数据

按照本问题上述数据预处理部分筛掉零值过多和最近未发货的运输线路并利用箱线图剔除异常值后，发现依旧存在货量较为特殊的运输线路，这类运输线路的特点为在 2021-01-01 至 2022-12-31 期间前段时间存在较大运货量，中间一段时间运货量为零，后面一段时间又逐渐恢复运货且最近货量一直保持正数。结合现实情况理解为此类运输线路收到了疫情等因素的较长时间影响，但近期物流场地和运输线路恢复。为此对数据进行进一步剔除，经过不断实践，最终选择剔除小于 3%最大值的数据作为处理方法，方法如下：

计算$DCi \to DCj$运输线路$\left(x_{i,j,1}, x_{i,j,2}, \cdots, x_{i,j,n}\right)$的最大值并将其记为$x_{i,j,max}$，剔除运输线路中所有货量小于$0.03x_{i,j,max}$的日期的数据。这样进一步剔除数据的具有如下优点：对于上述特殊的运输线路，可以剔除疫情等因素对预测的影响，使得预测更精确。而对于一直平稳运营的运输线路，仅剔除货量小于$0.03x_{i,j,max}$只剔除了部分数据点，对预测影响不大。

4.3.4 分析平稳性并差分处理不平稳数据

本问题中，称可达值$d_{i,j} \neq 0$的运输线路剔除异常值后的数据为有效数据，记由物流场地i到物流场地j的运输线路的第k个有效数据为$x_{i,j,k}$，其中$k = 1,2,\cdots,n$，对数据进行平稳性检验。

记$x_{i,j,k}$的秩为$R_{i,j,k} = R\left(x_{i,j,k}\right)$，首先计算变量$(t, R_t)$的$Spearman$相关系数$q_s$如式(4.2)，它是衡量两个变量依赖性的非参数指标[1]。其中n为有效数据个数。

$$q_s = 1 - \frac{6}{n(n^2-1)}\sum_{k=1}^{n}\left(t - R_{i,j,k}\right)^2 \tag{4.2}$$

构造统计量T：

$$T = \frac{q_s\sqrt{n-2}}{\sqrt{1-{q_s}^2}} \tag{4.3}$$

其中n为有效数据个数。

由数理统计中假设检验的理论知：当$|T| < t_\alpha(n-2)$时，序列具有平稳性；当$|T| \geq t_\alpha(n-2)$时，序列不平稳。其中$t_\alpha(n-2)$是t分布自由度为$n-2$的α分位数。

当序列不平稳时，需要对数据首先进行差分，再利用差分后的平稳数据进行预测。本问题中，因为不平稳数据经过一次差分后就平稳了，为避免因不必要的差分造成时间序列预测模型扭曲或预测精度降低，因此选用一次差分。设差分后的数据为$\triangle x_{i,j,k}$，则$\triangle x_{i,j,k}$与$x_{i,j,k+1}$、$x_{i,j,k}$有如下关系：

$$\triangle x_{i,j,k} = x_{i,j,k+1} - x_{i,j,k} \tag{4.4}$$

4.4 基于$ARMA$的货量预测模型

为对 2023 年 1 月 1 日到 2023 年 1 月 31 日运输线路的货量进行预测，本问题基于$ARMA$时间序列预测构建运输线路货量预测模型。

定义白噪声$\varepsilon_{i,j,t}$：

$$\varepsilon_{i,j,t} = x_{i,j,k} - \frac{\sum_{t=1}^{n} x_{i,j,t}}{n} \tag{4.5}$$

定义后移算子$\mathcal{B}$，满足式(4.7)：

$$\mathcal{B}x_{i,j,k} = x_{i,j,k-1} \tag{4.7}$$

构建货量预测模型如下：

$$x_{i,j,t} = \Phi_0 + \Phi_1\mathcal{B}x_{i,j,t} + \cdots + \Phi_p\mathcal{B}^p x_{i,j,t} - \theta_1\mathcal{B}\varepsilon_{i,j,t} - \cdots - \theta_q\mathcal{B}^q\varepsilon_{i,j,t} \tag{4.8}$$

其中，$\Phi_p \neq 0$，$\theta_q \neq 0$，p为自回归序列阶数，q为滑动平均序列阶数。

为预测物流场地i到物流场地j的 2023 年 1 月 1 日到 2023 年 1 月 31 日货量，下需求解系数向量$(\Phi_0, \Phi_1, \cdots, \Phi_p, \theta_1, \cdots, \theta_q)$，可以利用最小二乘法进行求解，求解公式为：

$$\begin{pmatrix} \Phi_0 \\ \Phi_1 \\ \vdots \\ \Phi_p \\ \theta_1 \\ \vdots \\ \theta_q \end{pmatrix} = [(X\Sigma)^T(X\Sigma)]^{-1} \cdot (X\Sigma)^T \begin{pmatrix} x_{i,j,1} \\ x_{i,j,2} \\ x_{i,j,3} \\ \vdots \\ x_{i,j,n-1} \\ x_{i,j,n} \end{pmatrix} \tag{4.9}$$

其中$X = \begin{pmatrix} 1 & x_{i,j,p} & x_{i,j,p+1} & \cdots & x_{i,j,n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{i,j,1} & x_{i,j,2} & \cdots & x_{i,j,n-1-p} \end{pmatrix}$，$\Sigma = \begin{pmatrix} \varepsilon_{i,j,q} & \varepsilon_{i,j,q-1} & \cdots & \varepsilon_{i,j,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \varepsilon_{i,j,1} & \varepsilon_{i,j,1} & \cdots & \varepsilon_{i,j,n-1-q} \end{pmatrix}$.

将系数向量代入模型公式(4.8), 对验证集进行验证，观察预测准确率。若准确率较高则利用此系数向量对应的模型公式(4.8)预测 2023 年 1 月 1 日到 2023 年 1 月 31 日运

输线路的货量。

4.5 问题 1 的模型求解

4.5.1 数据预处理

以$DC14 \rightarrow DC10$、$DC20 \rightarrow DC35$、$DC25 \rightarrow DC62$三条运输线路为例，其他运输线路处理方法类似。图 4.2 给出了 2021-01-01 至 2022-12-31 期间$DC14 \rightarrow DC10$、$DC20 \rightarrow DC35$、$DC25 \rightarrow DC62$这 3 条运输线路货量未进行数据清洗的历史数据。可以观察到未进行数据清洗时的货量历史数据具有较多零值和异常值，对预测结果的准确性影响较大。

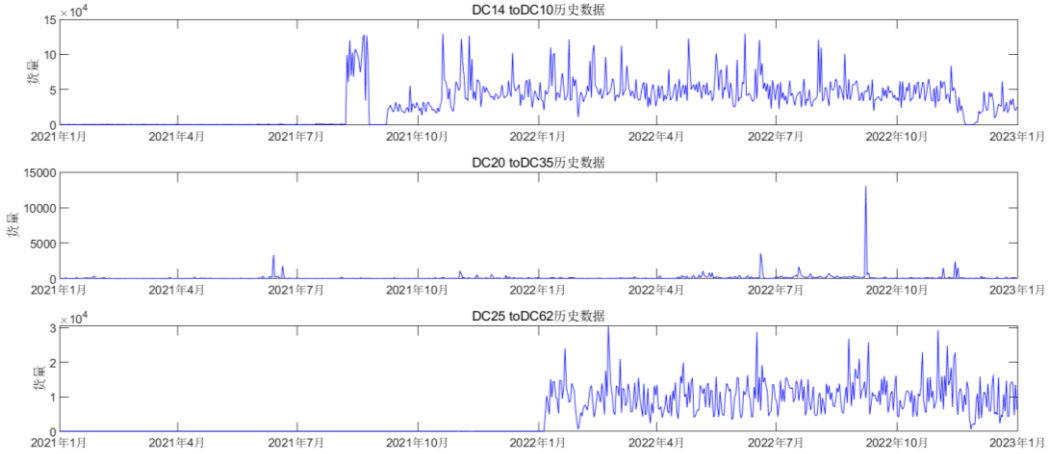

图 4.2　2021-01-01 至 2022-12-31 期间 3 条运输线路货量历史数据（未清洗）

图 4.3 给出了 2021-01-01 至 2022-12-31 期间这 3 条运输线路货量进行筛掉零值过多和最近未发货的线路、箱线图剔除异常值、剔除小于 3%最大值处理后的数据。对数据预处理后得到的 503 条数据序列进行平稳性检验，发现其中 230 条通过平稳性检验，273 条未通过平稳性检验。对这 273 条数据进行一阶差分处理，差分后的数据通过了平稳性检验。由图 4.3 观察到进行了数据清洗后的货量数据呈现较为平稳的分布。

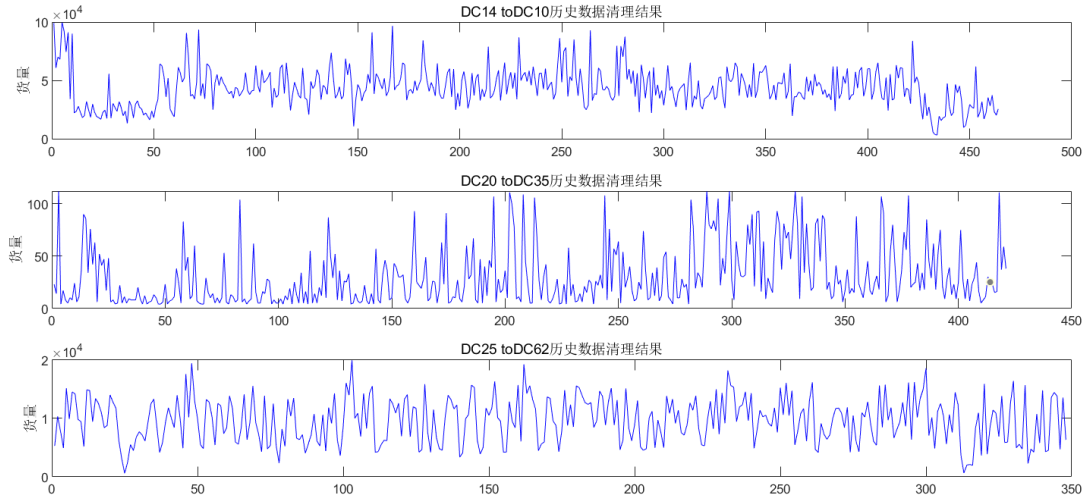

图 4.3　2021-01-01 至 2022-12-31 期间 3 条运输线路货量数据（已清洗）

利用$Spearman$相关系数法分析上述经过数据清洗后的数据的平稳性，若平稳则利用上述数据直接进行模型求解；若不平稳则还需对此部分数据进行差分处理，然后再利用差分后的数据进行求解。

4.5.2 货量预测模型求解

本问题求解过程中对每条运输线路分别进行预测，因此每条线路的求解方法类似，仍以$DC14 \to DC10$、$DC20 \to DC35$、$DC25 \to DC62$这 3 条运输线路的货量预测为例。

代入经过数据预处理的数据，利用$Matlab$编写程序，对参数p和q的值不断调试，发现当自回归序列阶数$p = 1$，滑动平均序列阶数$q = 5$时，预测误差较小，因此对本问题训练集利用构建的基于$ARMA(1,5)$货量预测模型进行求解。代入验证集对货量预测模型进行检验，这 3 条线路的相对误差的绝对值如表 1，此模型准确率较高，可以用来预测。

表 4.1 货量预测模型对验证集检验时货量的相对误差绝对值

运输线路	最大误差
$DC14 \to DC10$	15.5926%
$DC20 \to DC35$	0.06685%
$DC25 \to DC62$	12.3807%

表 4.2、表 4.3、表 4.4 分别给出了$DC14 \to DC10$、$DC20 \to DC35$、$DC25 \to DC62$在 2023-01-01 至 2023-01-31 期间每天货量预测值。

表 4.2 $DC14 \to DC10$在 2023-01-01 至 2023-01-31 期间每天货量预测值

日期	货量预测值	日期	货量预测值
2023-01-01	35034	2023-01-17	20763
2023-01-02	29458	2023-01-18	33580
2023-01-03	20290	2023-01-19	23289
2023-01-04	42793	2023-01-20	35220
2023-01-05	12319	2023-01-21	24284
2023-01-06	26218	2023-01-22	29350
2023-01-07	31248	2023-01-23	20103
2023-01-08	17834	2023-01-24	19974
2023-01-09	43727	2023-01-25	27816
2023-01-10	22424	2023-01-26	26916
2023-01-11	37439	2023-01-27	31688
2023-01-12	40340	2023-01-28	39710
2023-01-13	45990	2023-01-29	31741
2023-01-14	15451	2023-01-30	15848
2023-01-15	39545	2023-01-31	39666
2023-01-16	21294		

表 4.3　$DC20 \rightarrow DC35$在 2023-01-01 至 2023-01-31 期间每天货量预测值

日期	货量预测值	日期	货量预测值
2023-01-01	46	2023-01-17	30
2023-01-02	42	2023-01-18	24
2023-01-03	24	2023-01-19	53
2023-01-04	39	2023-01-20	45
2023-01-05	42	2023-01-21	66
2023-01-06	11	2023-01-22	44
2023-01-07	24	2023-01-23	40
2023-01-08	33	2023-01-24	65
2023-01-09	44	2023-01-25	48
2023-01-10	49	2023-01-26	64
2023-01-11	43	2023-01-27	77
2023-01-12	63	2023-01-28	80
2023-01-13	40	2023-01-29	77
2023-01-14	54	2023-01-30	58
2023-01-15	66	2023-01-31	83
2023-01-16	41		

表 4.4　$DC25 \rightarrow DC62$在 2023-01-01 至 2023-01-31 期间每天货量预测值

日期	货量预测值	日期	货量预测值
2023-01-01	8628	2023-01-17	8628
2023-01-02	11053	2023-01-18	11053
2023-01-03	6432	2023-01-19	6432
2023-01-04	8399	2023-01-20	8399
2023-01-05	6881	2023-01-21	6881
2023-01-06	7022	2023-01-22	7022
2023-01-07	8580	2023-01-23	8580
2023-01-08	9074	2023-01-24	9074
2023-01-09	5199	2023-01-25	5199
2023-01-10	10221	2023-01-26	10221
2023-01-11	11163	2023-01-27	11163
2023-01-12	7270	2023-01-28	7270
2023-01-13	9640	2023-01-29	9640
2023-01-14	7083	2023-01-30	7083
2023-01-15	9583	2023-01-31	9583
2023-01-16	9305		

利用$Matlab$绘制出$DC14 \rightarrow DC10$、$DC20 \rightarrow DC35$、$DC25 \rightarrow DC62$这 3 条运输线路在 2023-01-01 至 2023-01-31 期间货量预测数据图（如图 4.4），其中红线是预测数据，该图直观表示结果。

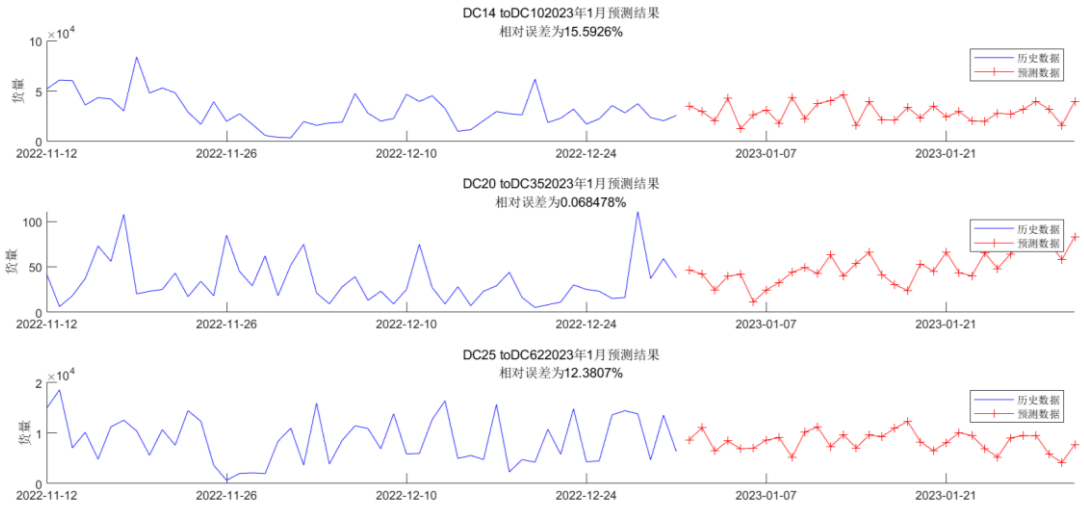

图 4.4　2023-01-01 至 2023-01-31 期间 3 条运输线路货量预测数据图

每条运输线路在 2023-01-01 至 2023-01-31 期间每天货量预测值数据在附件中。

五、问题 2 的模型建立与求解

5.1　问题 2 的分析

问题 2 是一个优化问题，本题需要在物流场地$DC5$关停时重新分配物流场地的发货量和收货量，首先考虑$DC5$相连的物流场地总发货量和总收货量尽可能不变，即所有货物尽量正常流转。其次在上述条件的基础上考虑改变线路尽可能少、改变线路负荷尽可能相同。

因此首先判断当$DC5$左相连运输线路（存在只经过一条运输线路就到达$DC5$的场地）和右相连运输线路（存在从$DC5$出发能够只经过一条运输线路就到达的场地）的运货量达到运输能力上限时，货物能否正常流转。正常流转即关停$DC5$后所有左相连的运输线路运货量总和高于需要运往$DC5$的货物量总和，所有右相连的运输线路运货量总和高于需要从$DC5$运出的货物量总和。若不能正常流转，则把所有与$DC5$相连线路的货量均调整为运输能力上限。若可以正常流转，则分别构建一个与$DC5$左相连的场地发货量的数学模型和一个与$DC5$右相连的场地收货量的数学规划模型。本问题的核心在构建这个具有两个目标函数的优化模型，考虑每个物流场地的处理能力和每条线路的运输能力，对目标函数加以约束并利用优化算法求解。

问题 2 流程图如图 5.1 所示。

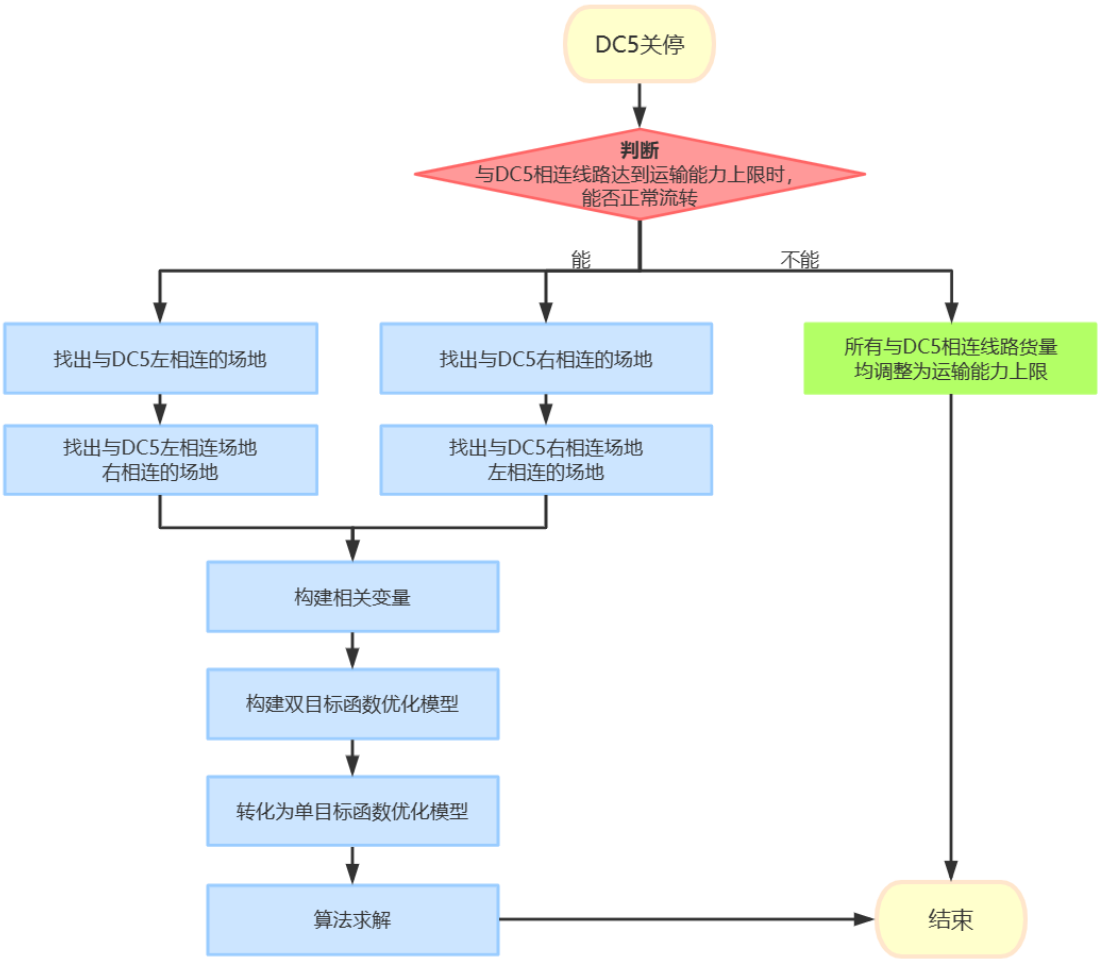

图 5.1　问题 2 流程图

5.2　模型准备

问题 2 中首先要求将物流场地$DC5$相关运输线路的货量分配到其他线路使所有货物尽量正常流转，因此本问题中考虑与$DC5$相连场地的运输线路运货量变化情况，具体是考虑与$DC5$左相连物流场地的发货量和右相连物流场地的收货量变化。

其中，与$DC5$左相连的物流场地指的是存在只经过一条运输线路就到达$DC5$的场地；与$DC5$右相连的物流场地指的是存在从$DC5$出发能够只经过一条运输线路就到达的场地。图解如图5.2。

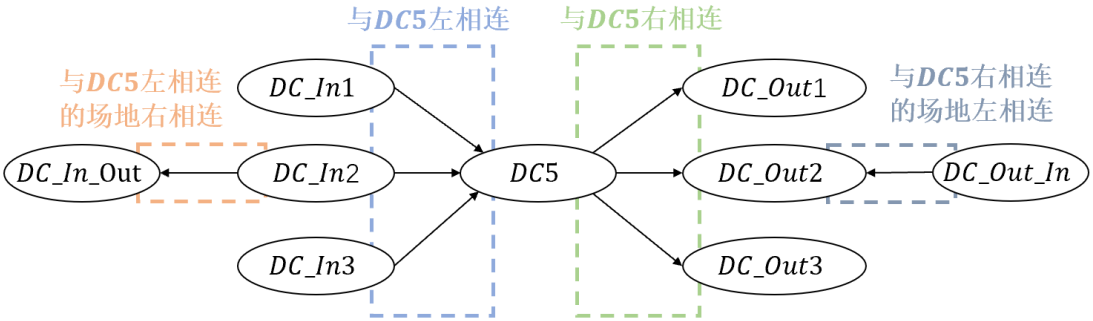

图 5.2　左相连、右相连定义图解

一、定义如下变量：

x_{ij}：$DC5$关停前从物流场地DCi到DCj运输线路的货量，若无线路$x_{ij}=0$；

W_1：物流场地$DC5$在某一天的总收货量，$W_1=\sum_{i=1}^{n}x_{i5}$；

W_2：物流场地$DC5$在某一天的总发货量，$W_2=\sum_{i=1}^{n}x_{5j}$；

x'_{ij}：$DC5$关停调整后从DCi到DCj运输线路的货量；

x_{ijt}：历史第t天从物流场地DCi到物流场地DCj的货量，其中$i,j=1,2,\cdots,81$，$t=1,2,\cdots 730$。

$M_{i\cdot}$：发货的场地DCi历史上最大发货量，即物流场地处理能力上限，满足

$$M_{i\cdot}=\max_{t}\sum_{j=1}^{81}x_{ijt} \tag{5.1}$$

$M_{\cdot j}$：收货的场地DCj历史上最大收货量，即物流场地处理能力上限，满足

$$M_{\cdot j}=\max_{t}\sum_{i=1}^{81}x_{ijt} \tag{5.2}$$

M_{ij}：从DCi到DCj运输线路的运输能力，即历史货量最大值。

k_{ij}：运输线路货量变化示性变量，满足

$$k_{ij}=\begin{cases}0，\ x'_{ij}=x_{ij}\\ 1，\ x'_{ij}>x_{ij}\end{cases} \tag{5.3}$$

二、定义如下集合：

与$DC5$左相连的物流场地集合E_1：$E_1=\{DCi|x_{i5}\neq 0\}$；

与$DC5$右相连的物流场地集合E_2：$E_2=\{DCj|x_{5j}\neq 0\}$；

与$DC5$左相连场地右相连的物流场地集合E_3（除$DC5$外）：

$$E_3=\{DCj|DCi\in E_1\text{且}x_{ij}\neq 0\}-\{DC5\}$$

与$DC5$右相连场地左相连的物流场地集合E_4（除$DC5$外）：

$$E_4=\{DCi|DCj\in E_2\text{且}x_{ij}\neq 0\}-\{DC5\}$$

物流场地集合E_1与E_3相连的有向线路构成的集合E'_1，即与$DC5$左相连的物流场地与其右相连场地之间的运输线路的集合：

$$E'_1=\{(DCi,DCj)|DCi\in E_1，\ DCj\in E_3\text{且}x_{ij}\neq 0\}$$

物流场地集合E_2与E_4相连的有向线路构成的集合E'_2，即与$DC5$右相连场地集合中的物流场地与其左相连场地之间的运输线路的集合：

$$E'_2=\{(DCj,DCi)|DCj\in E_2，\ DCi\in E_4\text{且}x_{ij}\neq 0\}$$

5.3 模型建立

5.3.1 判断能否正常流转

正常流转即关停$DC5$后所有左相连的运输线路运货量总和高于需要运往$DC5$的货物量总和，所有右相连的运输线路运货量总和高于需要从$DC5$运出的货物量总和。

计算式(5.4)和式(5.5)：

$$\Delta W_1 = \sum_{\substack{DCi\in E_1 \\ DCj\in E_3}} \left(M_{ij} - x_{ij}\right) - W_1 \tag{5.4}$$

$$\Delta W_2 = \sum_{\substack{DCj\in E_2 \\ DCi\in E_4}} \left(M_{ij} - x_{ij}\right) - W_2 \tag{5.5}$$

其中ΔW_1、ΔW_2的值的正负有如下四种关系，代表四种情况：

I. $\Delta W_1 \geq 0$且$\Delta W_2 \geq 0$

所有相连运输线路的运货量达到运输能力上限时高于需要运往$DC5$和从$DC5$运出的货量总和，货物能正常流转；

II. $\Delta W_2 \geq 0$但$\Delta W_1 < 0$

即使所有与$DC5$左相连物流场地与其右相连场地之间的运输线路运货量达到运输能力上限，还会有一部分货物无法正常流转。

III. $\Delta W_1 \geq 0$但$\Delta W_1 < 0$

即使所有与$DC5$右相连物流场地与其左相连场地之间的运输线路运货量达到运输能力上限，还会有一部分货物无法正常流转。

IV. $\Delta W_1 < 0$且$\Delta W_2 < 0$

情况II和情况III均发生。

5.3.2 调整到上限无法满足损失货量时的分流方案

当$\Delta W_1 < 0$或$\Delta W_2 < 0$时，以使货物尽量正常流转作为决策的第一原则，即将所有与$DC5$左相连（$\Delta W_1 < 0$）或右相连（$\Delta W_2 < 0$）线路的货量均调整为运输能力上限，这样才能使不能正常流转的货量最少。

一、$\Delta W_1 < 0$

考虑到当$\Delta W_1 < 0$时，无论如何增大与$DC5$左相连物流场地运输线路的货量，都无法达到关停$DC5$损失的收货量，因此令与$DC5$左相连的物流场地集合E_1的所有除$DC5$外右相连物流场地的运输线路运货量调整到上限，即$x'_{ij} = M_{ij}$，其中i满足$DCi \in E_1$，j满足$DCj \in E_3$。

二、$\Delta W_2 < 0$

同理，当$\Delta W_2 < 0$时现需要令与$DC5$右相连的物流场地集合E_2的所有除$DC5$外左相连物流场地的运输线路运货量调整到上限，即$x'_{ij} = M_{ij}$，其中j满足$DCj \in E_2$，i满足$DCi \in E_4$。

5.3.3 达到损失发/收货量时的分流方案

当$\Delta W_1 \geq 0$或$\Delta W_2 \geq 0$时，说明有一些分流方案在关停$DC5$后能使得货物左相连场地的货物或右相连场地的货物正常流转。

一、$\Delta W_1 \geq 0$

考虑到当$\Delta W_1 \geq 0$时，调整后的与$DC5$左相连物流场地右相连运输线路的货量可以达到关停$DC5$损失的收货量。因此构造优化模型如下：

（一）构造约束条件

1. 达到关停$DC5$损失的收货量

$DC5$关停调整后左相连场地集合中的场地到其右相连物流场地运输线路的发货量达到了$DC5$关停损失的收货量。

$$\sum_{\substack{DCi\in E_1 \\ DCj\in E_3}} (x'_{ij} - x_{ij}) = W_1 \tag{5.6}$$

2. 修改后的运输线路货量不能比修改前的小，且不能比历史货量最大值大

$$x_{ij} \leq x'_{ij} \leq M_{ij} \tag{5.7}$$

其中$DCi \in E_1$，$DCj \in E_3$。

3. 每个物流场地的发货或收货量不超过物流场地的处理能力

在本题中，将物流场地的处理能力定义为场地历史发货或收货的货量最大值，故此约束条件可表示为如下两个式子：

$$\sum_{DCj\in E_3} x'_{ij} \leq M_{i\cdot}, \ \forall DCi \in E_1 \tag{5.8}$$

$$\sum_{DCi\in E_1} (x'_{ij} - x_{ij}) \leq M_{\cdot j} - \sum_{i=1}^{n} x_{ij}, \ \forall DCj \in E_3 \tag{5.9}$$

（二）构造目标函数

1. 运货量发生改变的运输线路数量尽可能少：

$$K_1 = \sum_{\substack{DCi\in E_1 \\ DCj\in E_3}} k_{ij} \tag{5.10}$$

其中$k_{ij} \in \{0,1\}$。

2. 从与$DC5$左相连场地集合到其右相连场地的运输线路工作负荷方差尽可能少：

$$K_2 = \sum_{\substack{DCi \in E_1 \\ DCj \in E_3}} \frac{\left(\frac{x'_{ij}}{M_{ij}} - u\right)^2}{|E'_1|} \tag{5.11}$$

其中，u表示运货量发生改变的运输线路的工作负荷增大的平均值，计算公式为：

$$u = \sum_{\substack{DCi \in E_1 \\ DCj \in E_3}} \frac{\frac{x'_{ij}}{M_{ij}}}{|E'_1|} \tag{5.12}$$

（三）带约束条件的双目标函数优化模型

因此，带约束条件的双目标函数优化模型如下：

$$\begin{aligned} &\min K_1 \\ &\min K_2 \\ &\text{s.t.} \begin{cases} \sum_{\substack{DCi \in E_1 \\ DCj \in E_3}} (x'_{ij} - x_{ij}) = W_1 \\ \sum_{DCj \in E_3} x'_{ij} \le M_{i\cdot}\ ,\ \forall DCi \in E_1 \\ \sum_{DCi \in E_1} (x'_{ij} - x_{ij}) \le M_{\cdot j} - \sum_{i=1}^{n} x_{ij}\ ,\ \forall DCj \in E_3 \\ x_{ij} \le x'_{ij} \le M_{ij},\ DCi \in E_1, DCj \in E_3 \\ K_1 = \sum_{\substack{DCi \in E_1 \\ DCj \in E_3}} k_{ij}\ ,\ k_{ij} \in \{0,1\} \\ K_2 = \sum_{\substack{DCi \in E_1 \\ DCj \in E_3}} \frac{\left(\frac{x'_{ij}}{M_{ij}} - u\right)^2}{|E'_1|} \\ u = \sum_{\substack{DCi \in E_1 \\ DCj \in E_3}} \frac{\frac{x'_{ij}}{M_{ij}}}{|E'_1|} \end{cases} \end{aligned} \tag{5.13}$$

（四）双目标优化转化为单目标优化

考虑到可以利用式子的最小值等于这个式子倒数的最大值，即

$$\begin{cases} \min K_1 = \min \sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} k_{ij} = \max \dfrac{1}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} k_{ij}} \\ \min K_2 = \min \sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} \dfrac{\left(\dfrac{x'_{ij}}{M_{ij}} - u\right)^2}{|E'_1|} = \max \dfrac{|E'_1|}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} \left(\dfrac{x'_{ij}}{M_{ij}} - u\right)^2} \end{cases} \tag{5.14}$$

因为两个目标函数都要取最大值，因此可以用乘积最大值表示，故可将目标函数转化为如下形式：

$$\max \frac{1}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} k_{ij}} \times \max \frac{|E'_1|}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} \left(\dfrac{x'_{ij}}{M_{ij}} - u\right)^2} = \max \frac{1}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} k_{ij}} \frac{|E'_1|}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} \left(\dfrac{x'_{ij}}{M_{ij}} - u\right)^2} \tag{5.15}$$

故$\Delta W_1 \geq 0$时，优化模型为：

$$\max \frac{1}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} k_{ij}} \frac{|E'_1|}{\sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} \left(\dfrac{x'_{ij}}{M_{ij}} - u\right)^2}$$

$$\text{s.t.} \begin{cases} \sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} (x'_{ij} - x_{ij}) = W_1 \\ \sum\limits_{DCj\in E_3} x'_{ij} \leq M_{i\cdot}\ ,\ \ \forall DCi \in E_1 \\ \sum\limits_{DCi\in E_1} (x'_{ij} - x_{ij}) \leq M_{\cdot j} - \sum\limits_{i=1}^{n} x_{ij}\ ,\ \ \forall DCj \in E_3 \\ x_{ij} \leq x'_{ij} \leq M_{ij},\ \ DCi \in E_1, DCj \in E_3 \\ u = \sum\limits_{\substack{DCi\in E_1 \\ DCj\in E_3}} \dfrac{\dfrac{x'_{ij}}{M_{ij}}}{|E'_1|} \\ k_{ij} \in \{0,1\} \end{cases} \tag{5.16}$$

二、$\Delta W_2 \geq 0$

考虑到当$\Delta W_2 \geq 0$时，调整后的与$DC5$右相连物流场地左相连运输线路的货量可以达到关停$DC5$损失的发货量。用和$\Delta W_1 \geq 0$同样的思路建立优化模型，只需物流场地所属的集合，因此建立如下优化模型：

$$\max \frac{1}{\sum\limits_{\substack{DCj\in E_2 \\ DCi\in E_4}} k_{ij}} \frac{|E'_2|}{\sum\limits_{\substack{DCj\in E_2 \\ DCi\in E_4}} \left(\dfrac{x'_{ij}}{M_{ij}} - u\right)^2} \tag{5.17}$$

$$\text{s.t.}\begin{cases}\sum\limits_{\substack{DCj\in E_2\\DCi\in E_4}}(x'_{ij}-x_{ij})=W_2\\ \sum\limits_{DCi\in E_4}x'_{ij}\le M_{\cdot j},\ \forall DCj\in E_2\\ \sum\limits_{DCj\in E_2}(x'_{ij}-x_{ij})\le M_{i\cdot}-\sum\limits_{j=1}^{n}x_{ij},\ \forall DCi\in E_4\\ x_{ij}\le x'_{ij}\le M_{ij},\ DCj\in E_2, DCi\in E_4\\ u=\sum\limits_{\substack{DCj\in E_2\\DCi\in E_4}}\dfrac{\frac{x'_{ij}}{M_{ij}}}{|E'_2|}\\ k_{ij}\in\{0,1\}\end{cases}\tag{5.18}$$

利用优化算法求解变量k_{ij}，x'_{ij}即得到分流方案。

5.4 问题 2 的模型求解

考虑到要求解两组变量k_{ij}、x'_{ij}，即首先判断是否分配到$DCi\to DCj$，再考虑每条线路分配的运货量，因此运用遗传算法求解示性变量k_{ij}时，运用模拟退火算法求解x'_{ij}。

5.4.1 遗传算法简介

遗传算法的本质是通过群体搜索技术，根据适者生存的原则逐代进化，最终得到最优解的算法[2]。遗传算法实现方法如下：

1. 确定问题可行解域和编码方法，把每个解利用编码方法表示出来；
2. 以目标函数作为适应度函数；
3. 初始化参数：种群大小M，最大迭代次数G，交叉方法和变异率p_m；
4. 初始种群，给出比较好的可行解作为亲代；
5. 不断进行交叉、变异操作，根据适应度函数择优选取M个新的亲代生存并进行下一次交叉、变异、择优操作，直至选出最优解。

遗传算法求解流程图如下：

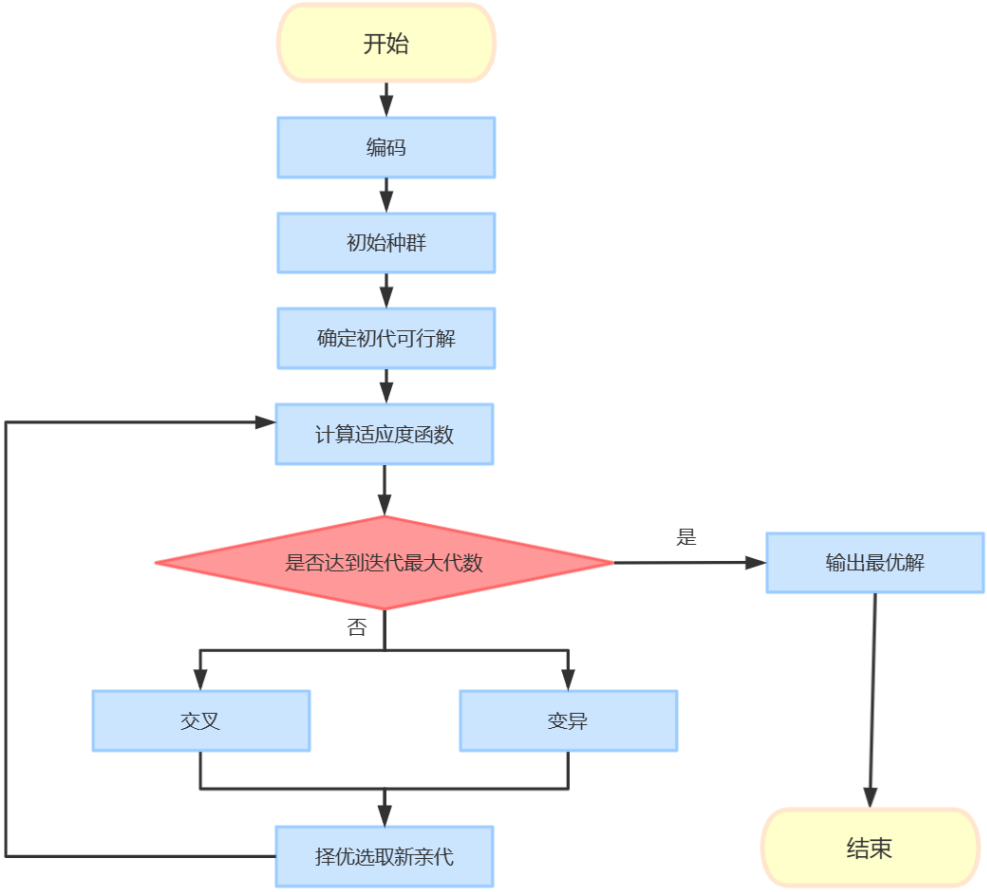

图 5.6　遗传算法流程图

5.4.2　模拟退火算法简介

模拟退火算法(SA)是以一种通过控制温度下降模拟物理退火过程来寻找最优值的优化算法。

模拟退火实现方法如下：

1. 初始化参数：初始温度T_0，降温系数α，最大迭代次数N，终止温度T_{end}，目标函数f；
2. 给出初始解ω，计算初始解的目标函数值$f(\omega)$；
3. 判断迭代次数是否达到最大迭代次数N，达到则结束并输出最优解，未达到则继续；
4. 扰动产生新解ω'，计算新解的目标函数$f(\omega')$；
5. 计算$\Delta f = f(\omega') - f(\omega)$，为寻找使得目标函数最大的最优解，$\Delta f \geq 0$时接受新解，$\Delta f < 0$时按$Metropolis$准则接受新解；
6. 判断是否达到终止温度，若达到则结束并输出最优解，若未达到则重复步骤 4 和 5。

其中，$Metropolis$准则是指

模拟退火算法求解流程图如下：

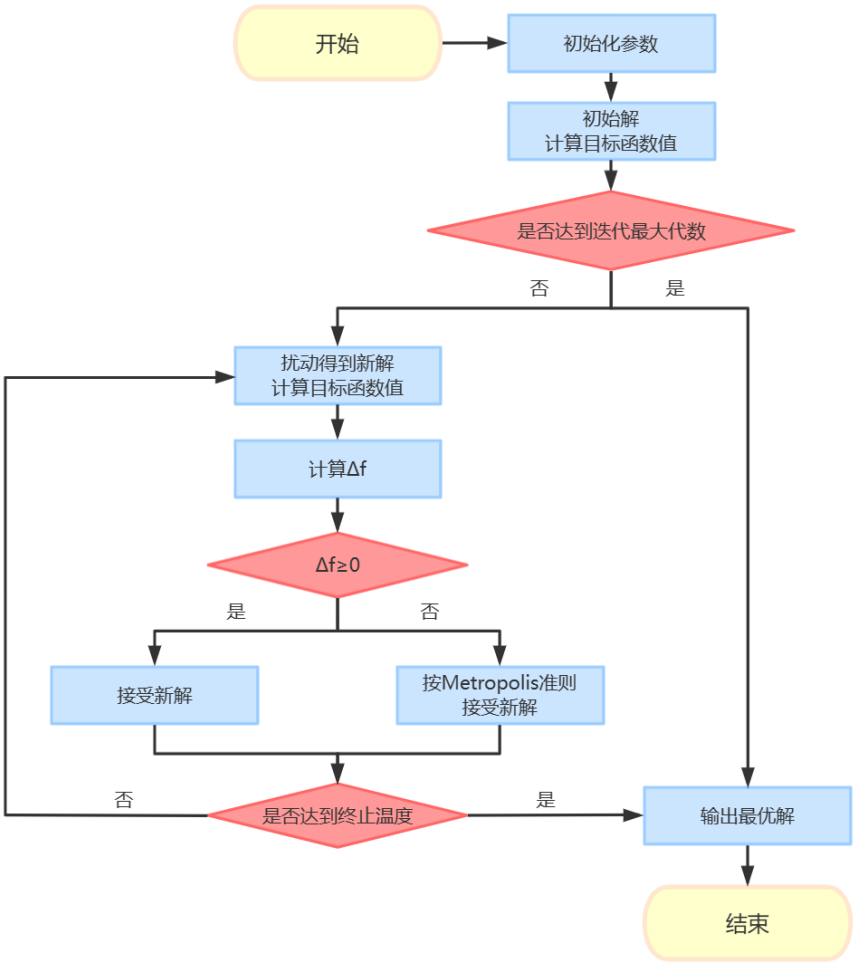

图 5.7　模拟退火算法流程图

5.4.3　求解结果

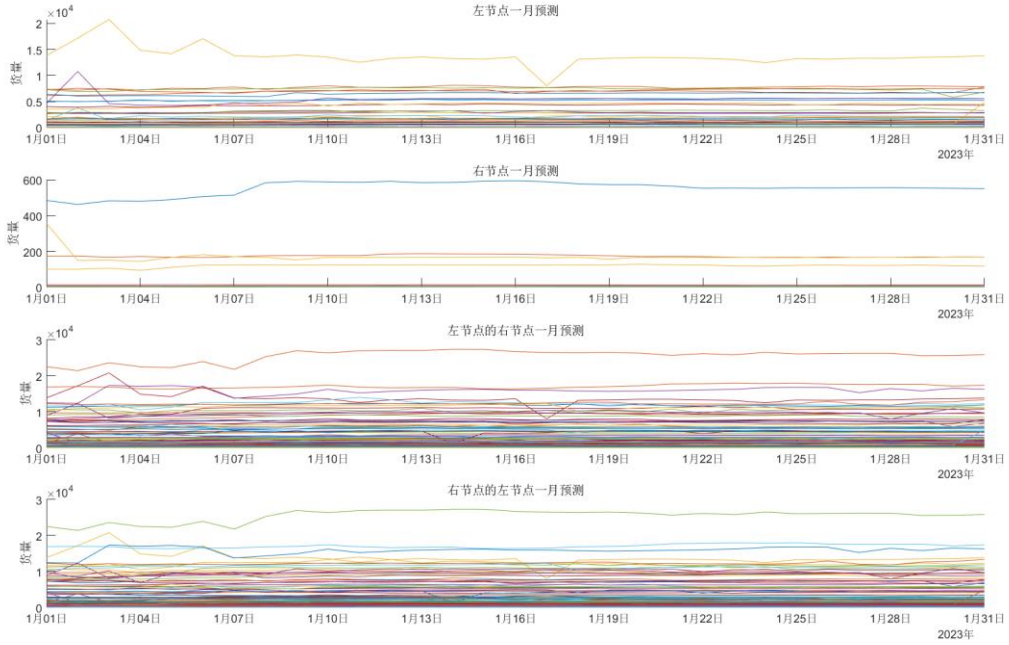

图 5.8　2023 年 1 月与$DC5$相连相关场地预测结果图

观察2023年1月与$DC5$相连相关场地预测结果如图5.8，问题2中遗传算法种群大小$M=40$，最大迭代次数$G=200$，交叉方法为三交换生成变异点，变异方法为单点变异其中变异率$p_m=0.1$；模拟退火初始温度$T_0=100℃$，降温系数$\alpha=0.9$，最大迭代次数$N=500$，终止温度$T_{end}=1℃$，目标函数f为问题2优化模型的目标函数。经过遗传算法和模拟退火算法相结合计算最优分流方案，2023年1月每天的方案输出在附件中。

2023年1月关停$DC5$后包裹都能正常流转，在单目标优化模型中，对于与$DC5$左相连场地右相连场地的数据，目标函数平均值为0.0368，平均货量发生变化的线路数有302条，网络负荷情况（用方差表示）约为8.9%。对于与$DC5$右相连场地左相连场地的数据，平均货量发生变化的线路数有254条，网络负荷情况（用方差表示）约为22.2%。普遍来讲，与$DC5$左相连场地右相连场地的目标函数比与$DC5$右相连场地左相连的目标函数大，这两个集合平均每天约有250条变动的线路重合。其中与$DC5$左相连场地与右相连场地的流量分配结果如图5.8、图5.9所示。

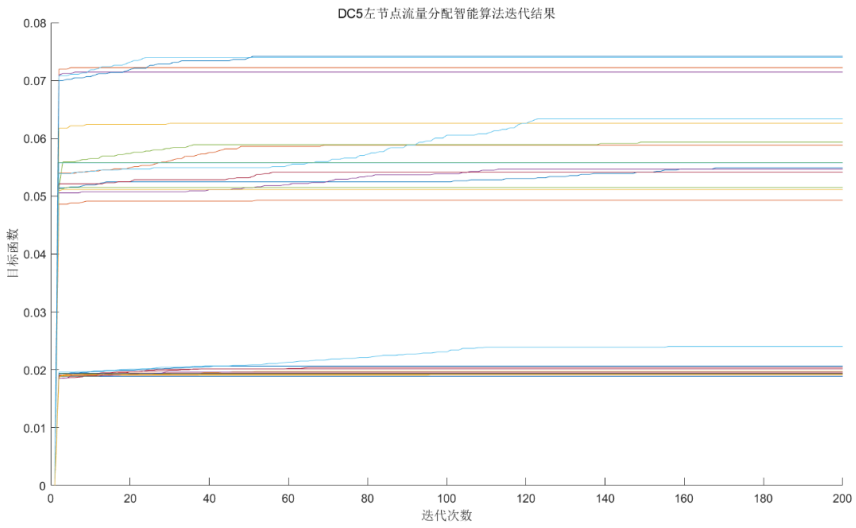

图5.9　$DC5$左相连场地流量分配结果图

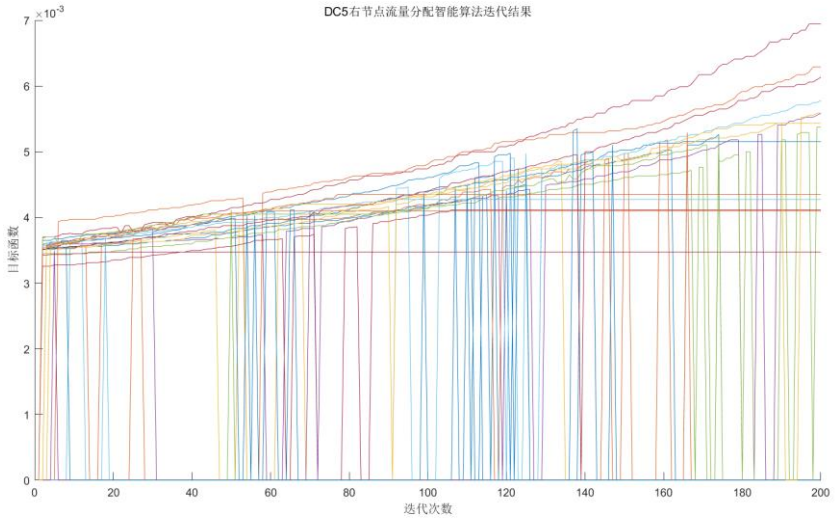

图5.10　$DC5$右相连场地流量分配结果图

六、问题 3 的模型建立与求解

6.1 问题 3 的分析

问题 3 也是一个优化问题，可以视为问题 2 的进一步讨论。与问题 2 不同的是，需要在物流场地$DC9$关停的情况下重新分配物流场地的发货量和收货量，除了可以调整已有线路的运货量，还可以开辟新的线路或者关闭已有线路，使得在改变线路尽可能少的情况、改变线路负荷尽可能一样的情况下，与$DC9$相连的物流场地总发货量和总收货量尽可能不变。问题 3 相比于问题 2 需要进一步讨论开辟或关闭的线路是哪些条，以及新开辟线路运货量为多少。

可以沿用问题 2 的思路解决问题 3，首先和问题 2 分析步骤一致，需要判断在开设运输线路数量达到上限且运输线路货量达到上限的情况下，能否达到因关停$DC9$损失的货量。若不能达到损失货量，则将所有线路调整为货量上限。若能够达到损失货量，则需确定最优决策方案，使得在达到因关停$DC9$损失的货量的前提下尽可能满足问题 3 题目的要求。同问题 2 一样，也需要将$DC9$的收货量和发货量分开讨论。

问题 3 流程图如图 6.1。

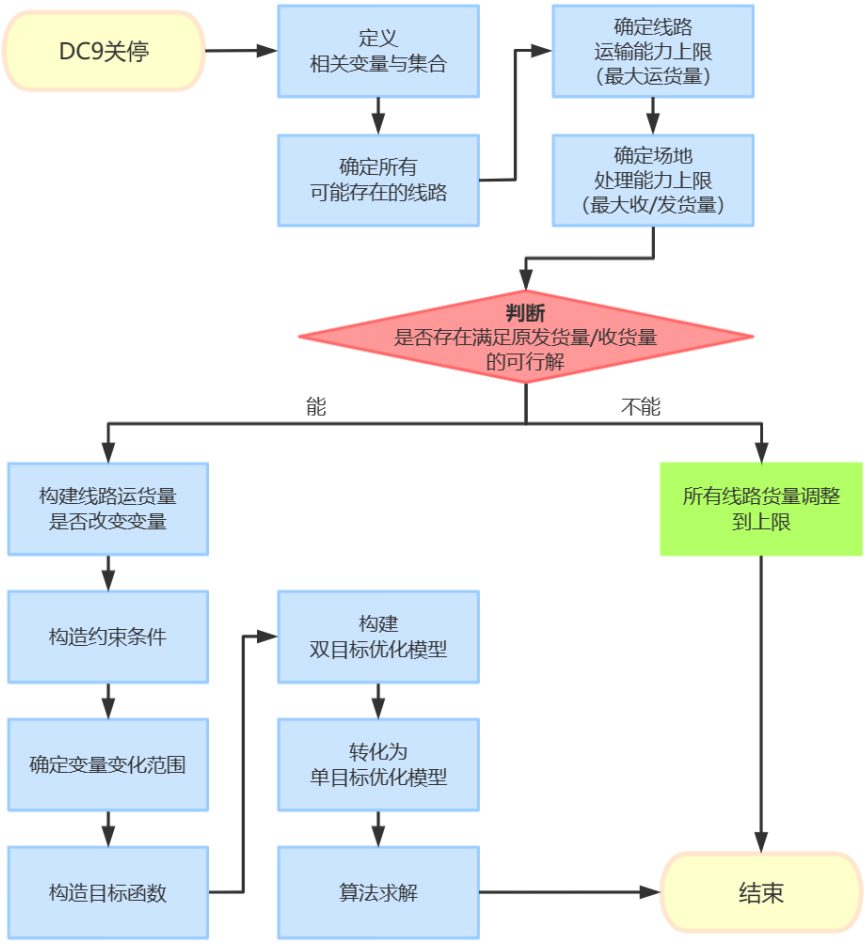

图 6.1　问题 3 流程图

6.2 模型准备

在判断在开设运输线路数量达到上限且运输线路货量达到上限的情况下，能否达到因关停$DC9$损失的货量之前，由于实际情况下物流场地之间存在距离，因此不是所有物流场地之间都可以开放线路，本问题中考虑只有历史上存在货流量的线路才可以开放。首先定义如下变量和集合：

y_{ij}：$DC9$关停当天之前从物流场地DCi到DCj运输线路的货量，若无线路$y_{ij}=0$；

y'_{ij}：$DC9$关停当天调整后从DCi到DCj运输线路的货量，若无线路，则$y'_{ij}=0$，规定：

$$y'_{i9}=0，\ i=1,2,\cdots,81$$

$$y'_{9j}=0，\ j=1,2,\cdots,81$$

x_{ijt}：历史第t天从物流场地DCi到物流场地DCj的货量，其中$i,j=1,2,\cdots,81$，$t=1,2,\cdots 730$。

F_1：关闭当天与$DC9$左相连的物流场地集合，$F_1=\{DCi|y_{i9}\neq 0\}$；

F_2：关闭当天与$DC9$右相连的物流场地集合，$F_2=\{DCj|y_{9j}\neq 0\}$；

如果新开线路，根据问题 3 题目中要求考虑与$DC9$相关线路的货量分配问题。选择以集合F_1中的场地作为发货点新开线路，以集合F_2中的场地为收货点新开线路。问题转化为选取集合F_1和F_2中的哪些点新开线路，以及如何选取新开线路的另一个场地。观察到在 2021-01-01 至 2022-12-31 的历史数据中存在很多运货量不为零的天数很少的临时运输线路，基于此定义如下集合：

F_3：与$DC9$左相连的场地右相连的物流场地集合，满足：

$$F_3=\{DCj|DCi\in F_1，\ y_{ij}\neq 0\}-\{DC9\}；$$

F_4：与$DC9$右相连的场地左相连的物流场地集合，满足：

$$F_4=\{DCi|DCj\in F_2，\ y_{ij}\neq 0\}-\{DC9\}；$$

F_5：历史数据中与$DC9$左相连的场地临时右相连的场地，将其视作可以开放发货线路的场地，满足：

$$F_5=\{DCj|DCi\in F_1，\ 1\leq\sum_t sgn(x_{ijt})\leq 20\}-\{DC9\}；$$

F_6：历史数据中与$DC9$右相连的场地临时左相连的场地，将其视作可以开放收货线路的场地，满足：

$$F_5=\{DCj|DCi\in F_1，\ 1\leq\sum_t sgn(x_{ijt})\leq 20\}-\{DC9\}；$$

其中$sgn(x_{ijt})$表示符号函数，满足

$$sgn(x_{ijt})=\begin{cases}1，\ x_{ijt}>0\\0，\ x_{ijt}=0\\-1，\ x_{ijt}<0\end{cases}$$

考虑到可以关闭部分线路，因此定义如下集合：

F_3'：与$DC9$左相连场地可能右相连的所有物流场地的集合，满足：

$$F_3' = F_3 \cup F_5;$$

F_4'：与$DC9$右相连场地可能左相连的所有物流场地的集合，满足：

$$F_4' = F_4 \cup F_6;$$

定义分流方案可能存在的运输线路的集合：

F_1'：分流方案可能存在的发货线路，满足：

$$F_1' = \{(DCi, DCj) | DCi \in F_1,\ DCj \in F_3',\ y_{ij} \neq 0 \vee 1 \leq \sum_t sgn(x_{ijt}) \leq 20\};$$

F_2'：分流方案可能存在的收货线路，满足：

$$F_2' = \{(DCi, DCj) | DCi \in F_4',\ DCj \in F_2,\ y_{ij} \neq 0 \vee 1 \leq \sum_t sgn(x_{ijt}) \leq 20\};$$

6.3 模型建立

6.3.1 关闭或新开线路的候选场地选取

由模型准备部分变量和集合的定义，y_{ij}'的正负满足如下关系：

$y_{ij}' \geq 0$：当$\begin{cases} DCi \in F_1,\ DCj \in F_3',\ (DCi, DCj) \in F_1' \\ DCi \in F_4',\ DCj \in F_2,\ (DCi, DCj) \in F_2' \end{cases}$二者满足其一时成立

$y_{ij}' = 0$：当$\begin{cases} DCi \in F_1,\ DCj \in F_3',\ (DCi, DCj) \notin F_1' \\ DCi \in F_4',\ DCj \in F_2,\ (DCi, DCj) \notin F_2' \end{cases}$二者满足其一时成立

由此完成第一步关闭或新开线路候选场地选取。

6.3.2 判断能否正常流转

本部分构建判断关停$DC9$后，在开设运输线路数量达到上限且运输线路货量达到上限的情况下，能否达到因关停$DC9$损失的货量的准则。

定义变量V_1、V_2如下：

$$V_1 = \sum_{\substack{DCi \in E_1 \\ DCj \in E_3 \cup \{9\}}} y_{ij} \tag{6.1}$$

$$V_2 = \sum_{\substack{DCi \in E_4 \\ DCj \in E_2 \cup \{9\}}} y_{ij} \tag{6.2}$$

分流方案需首先满足使得集合F_1中的场地的总发货量尽可能达到V_1，集合F_2中的场地的总发货量尽可能达到V_2。因此首先判断是否存在分流方案满足上述条件，为此定义如下变量：

M_{ij}：运输线路$DCi \to DCj$的最大运货量，满足：

$$M_{ij}=\begin{cases}\max\limits_{t} x_{ijt}\ ,\ y_{ij}\neq 0\\ \max\limits_{i,j}\max\limits_{t} x_{ijt}\ ,\ y_{ij}=0\end{cases} \tag{6.3}$$

其中$DCi\in F_1$，$DCj\in F_3'$或$DCi\in F_4'$，$DCj\in F_2$。

$M_{i\cdot}$：物流场地DCi历史上最大发货量，满足

$$M_{i\cdot}=\max_{t}\sum_{j=1}^{n} x_{ijt} \tag{6.4}$$

$M_{\cdot j}$：物流场地DCj历史上最大收货量，满足

$$M_{\cdot j}=\max_{t}\sum_{i=1}^{n} x_{ijt} \tag{6.5}$$

计算式(6.6)和式(6.7)：

$$\Delta V_1=\sum_{(DCi,DCj)\in F_1'}\left(M_{ij}-y_{ij}\right)-V_1 \tag{6.6}$$

$$\Delta V_2=\sum_{(DCi,DCj)\in F_2'}\left(M_{ij}-y_{ij}\right)-V_2 \tag{6.7}$$

即判断所有可能发货的线路都存在且运货量达到上限时，是否可以达到$DC9$未停运时的左相连场地集合的总发货量。所有可能收货的线路都存在且运货量达到上限时，是否可以达到$DC9$未停运时的右相连节点的总收货量。

6.3.3 调整到上限无法满足损失货量时的分流方案

若$\Delta V_1<0$或$\Delta V_2<0$，则说明即使将可新开的发货场地或收货场地均开放，也无法达到因关停$DC9$所带来的货量损失量。这种情况与问题 2 的处理方法相类似，即在以满足发货量和收获量为第一准则的情况下，新开所有可以能够新开的运输路线，并使得其运货量达到最大。

6.3.4 达到损失发/收货量时的分流方案

当$\Delta V_1\geq 0$或$\Delta V_2\geq 0$时，说明有一些分流方案能达到因$DC9$关停损失的发货量或收货量损失。

一、$\Delta V_1\geq 0$

考虑到当$\Delta V_1\geq 0$时，调整后的与$DC9$左相连场地右相连运输线路的货量可以达到关停$DC9$损失的收货量。首先做出以下准备：

定义所有可开放的临时线路当前运货量为零：

$$y_{ij}=0，(DCi,DCj)\in F_1',DCj\in F_5$$

$$y_{ij}=0，(DCi,DCj)\in F_2',DCi\in F_6$$

定义线路运货量改变示性变量：

$$k_{ij}=\begin{cases}1，\ y'_{ij}\neq y_{ij},(DCi,DCj)\in F'_1\\0，\ y'_{ij}=y_{ij},(DCi,DCj)\in F'_1\end{cases} \tag{6.8}$$

因此构造优化模型如下：

（一）构造约束条件

1. 分流方案总发货量需达到$DC9$未关停时集合F_1中场地的总发货量：

$$\sum_{(DCi,DCj)\in F'_1} y'_{ij}=V_1 \tag{6.9}$$

2. 分流方案的每条线路发货量不能高于其运输能力上限即历史最大值，每个场地的发货量和收货量不能高于其处理能力上限即历史最大值：

$$0\le y'_{ij}\le M_{ij}，\ \forall(DCi,DCj)\in F'_1 \tag{6.10}$$

$$\sum_{DCi\in F_1}(y'_{ij}-y_{ij})\le M_{\cdot j}-\sum_{i=1}^{n}y_{ij}，\ \forall DCj\in F'_3 \tag{6.11}$$

$$\sum_{DCj\in F'_3} y'_{ij}\le M_{i\cdot}，\ \forall DCi\in F_1 \tag{6.12}$$

（二）构造目标函数

1. $DC9$关停前后运货量发生改变的运输线路数量尽可能少：

$$K'_1=\sum_{(DCi,DCj)\in F'_1} k_{ij} \tag{6.14}$$

其中$k_{ij}\in\{0,1\}$。

2. 分流方案调整后所有运输线路的工作负荷方差尽可能少：

$$K'_2=\sum_{(DCi,DCj)\in F'_1}\frac{\left(\frac{y'_{ij}}{M_{ij}}-u\right)^2}{\sum_{(DCi,DCj)\in F'_1} sgn\left(y'_{ij}\right)} \tag{6.15}$$

其中，u表示运货量发生改变的运输线路的工作负荷增大的平均值，计算公式为：

$$u=\sum_{(DCi,DCj)\in F'_1}\frac{\frac{y'_{ij}}{M_{ij}}}{\sum_{(DCi,DCj)\in F'_1} sgn\left(y'_{ij}\right)} \tag{6.16}$$

（三）带约束条件的双目标函数优化模型

因此，带约束条件的双目标函数优化模型构建如下：

$$\min K'_1$$

$$\min K'_2$$

$$
\text{s.t.}\begin{cases}
\sum\limits_{(DCi,DCj)\in F_1'} y_{ij}' = V_1 \\
0 \le y_{ij}' \le M_{ij},\ \forall (DCi,DCj) \in F_1' \\
\sum\limits_{DCi\in F_1} (y_{ij}' - y_{ij}) \le M_{\cdot j} - \sum\limits_{i=1}^{n} y_{ij},\ \forall DCj \in F_3' \\
\sum\limits_{DCj\in F_3'} y_{ij}' \le M_{i\cdot},\ \forall DCi \in F_1 \\
K_1' = \sum\limits_{(DCi,DCj)\in F_1'} k_{ij} \\
K_2' = \sum\limits_{(DCi,DCj)\in F_1'} \dfrac{\left(\dfrac{y_{ij}'}{M_{ij}} - u\right)^2}{\sum_{(DCi,DCj)\in F_1'} sgn(y_{ij}')} \\
k_{ij} = \begin{cases} 1,\ y_{ij}' \ne y_{ij}, (DCi,DCj) \in F_1' \\ 0,\ y_{ij}' = y_{ij}, (DCi,DCj) \in F_1' \end{cases} \\
u = \sum\limits_{(DCi,DCj)\in F_1'} \dfrac{\dfrac{y_{ij}'}{M_{ij}}}{\sum_{(DCi,DCj)\in F_1'} sgn(y_{ij}')}
\end{cases} \tag{6.17}
$$

（四）双目标优化转化为单目标优化

为使其更便于计算机求解，对目标函数进行简化，将双目标优化模型转化为单目标优化模型。考虑到可以利用式子的最小值等于这个式子倒数的最大值，即

$$
\begin{cases}
\min K_1' = \min \sum\limits_{(DCi,DCj)\in F_1'} k_{ij} = \max \dfrac{1}{\sum_{(DCi,DCj)\in F_1'} k_{ij}} \\
\min K_2 = \min \sum\limits_{(DCi,DCj)\in F_1'} \dfrac{\left(\dfrac{y_{ij}'}{M_{ij}} - u\right)^2}{\sum_{(DCi,DCj)\in F_1'} sgn(y_{ij}')} = \max \dfrac{\sum_{(DCi,DCj)\in F_1'} sgn(y_{ij}')}{\sum_{(DCi,DCj)\in F_1'} \left(\dfrac{y_{ij}'}{M_{ij}} - u\right)^2}
\end{cases} \tag{6.18}
$$

因为两个目标函数都要取最大值，因此可以用乘积最大值表示。故$\Delta W_1 \ge 0$时，优化模型为：

$$
\max \frac{1}{\sum_{(DCi,DCj)\in F_1'} k_{ij}} \frac{\sum_{(DCi,DCj)\in F_1'} sgn(y_{ij}')}{\sum_{(DCi,DCj)\in F_1'} \left(\dfrac{y_{ij}'}{M_{ij}} - u\right)^2}
$$

$$
\text{s.t.}\begin{cases}
\sum\limits_{(DCi,DCj)\in F_1'} y_{ij}' = V_1 \\
0 \le y_{ij}' \le M_{ij}，\ \forall (DCi,DCj) \in F_1' \\
\sum\limits_{DCi\in F_1} (y_{ij}' - y_{ij}) \le M_{\cdot j} - \sum\limits_{i=1}^{n} y_{ij}，\ \forall DCj \in F_3' \\
\sum\limits_{DCj\in F_3'} y_{ij}' \le M_{i\cdot}，\ \forall DCi \in F_1 \\
K_1' = \sum\limits_{(DCi,DCj)\in F_1'} k_{ij} \\
K_2' = \sum\limits_{(DCi,DCj)\in F_1'} \dfrac{\left(\dfrac{y_{ij}'}{M_{ij}} - u\right)^2}{\sum_{(DCi,DCj)\in F_1'} sgn\left(y_{ij}'\right)} \\
k_{ij} = \begin{cases} 1，\ y_{ij}' \ne y_{ij}, (DCi,DCj) \in F_1' \\ 0，\ y_{ij}' = y_{ij}, (DCi,DCj) \in F_1' \end{cases} \\
u = \sum\limits_{(DCi,DCj)\in F_1'} \dfrac{\dfrac{y_{ij}'}{M_{ij}}}{\sum_{(DCi,DCj)\in F_1'} sgn\left(y_{ij}'\right)}
\end{cases} \tag{6.19}
$$

二、$\Delta V_2 \ge 0$

考虑到当$\Delta V_2 \ge 0$时，调整后的与$DC9$右相连物流场地运输线路的货量可以达到关停$DC9$损失的发货量。用和$\Delta V_1 \ge 0$同样的思路建立优化模型，只需物流场地所属的集合，因此建立如下优化模型：

$$
\max \frac{1}{\sum_{(DCi,DCj)\in F_2'} k_{ij}} \frac{\sum_{(DCi,DCj)\in F_2'} sgn\left(y_{ij}'\right)}{\sum_{(DCi,DCj)\in F_2'} \left(\dfrac{y_{ij}'}{M_{ij}} - u\right)^2}
$$

$$
\text{s.t.}\begin{cases}
\sum\limits_{(DCi,DCj)\in F_2'} y_{ij}' = V_2 \\
0 \le y_{ij}' \le M_{ij},\ \forall (DCi,DCj) \in F_2' \\
\sum\limits_{DCj\in F_2} (y_{ij}' - y_{ij}) \le M_{i\cdot} - \sum\limits_{j=1}^{n} y_{ij},\ \forall DCj \in F_4' \\
\sum\limits_{DCi\in F_4'} y_{ij}' \le M_{\cdot j},\ \forall DCj \in F_2 \\
K_1' = \sum\limits_{(DCi,DCj)\in F_2'} k_{ij} \\
K_2' = \sum\limits_{(DCi,DCj)\in F_2'} \dfrac{\left(\dfrac{y_{ij}'}{M_{ij}} - u\right)^2}{\sum_{(DCi,DCj)\in F_2'} sgn(y_{ij}')} \\
k_{ij} = \begin{cases} 1,\ y_{ij}' \ne y_{ij}, (DCi,DCj) \in F_2' \\ 0,\ y_{ij}' = y_{ij}, (DCi,DCj) \in F_2' \end{cases} \\
u = \sum\limits_{(DCi,DCj)\in F_2'} \dfrac{\dfrac{y_{ij}'}{M_{ij}}}{\sum_{(DCi,DCj)\in F_2'} sgn(y_{ij}')}
\end{cases} \tag{6.20}
$$

利用优化算法求解变量x_{ij}'，k_{ij}即得到分流方案。

6.4 问题 3 的模型求解

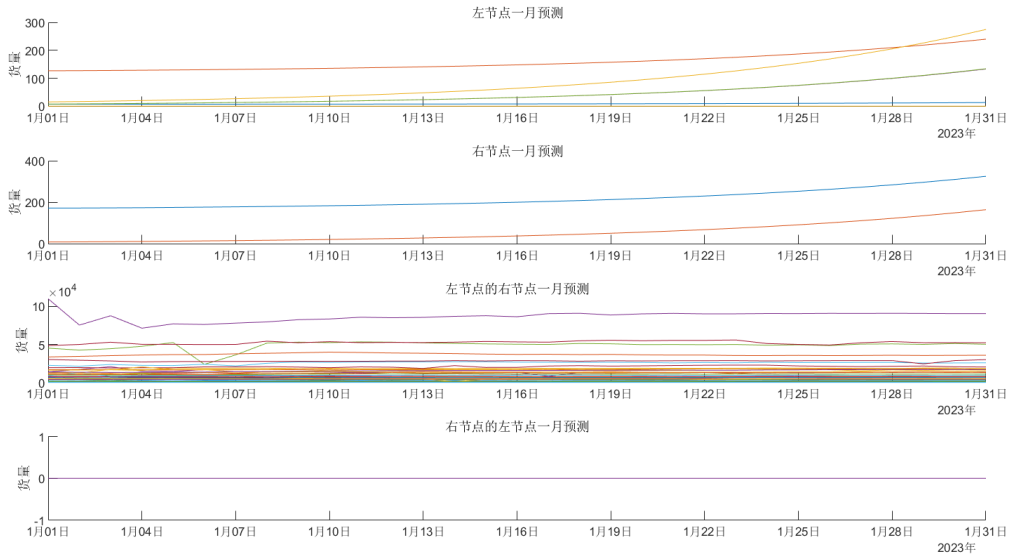

图 6.2　2023 年 1 月与$DC9$相连相关场地预测结果图

2023 年 1 月与$DC9$相连相关场地预测结果如图6.2，观察到与$DC9$左相连场地右相连的场地货量非常多，而与$DC9$右相连场地左相连的场地货量为 0 且 31 天内基本不变。

同样利用遗传算法与模拟退火算法相结合求解，问题 3 中基础参数设置和问题 2 一致，即遗传算法种群大小$M = 40$，最大迭代次数$G = 200$，交叉方法为三交换生成变异点，变异方法为单点变异其中变异率$p_m = 0.1$；模拟退火初始温度$T_0 = 100℃$，降温系数$\alpha = 0.9$，最大迭代次数$N = 500$，终止温度$T_{end} = 1℃$，目标函数f为问题 2 优化模型的目标函数。经过遗传算法和模拟退火算法相结合计算最优分流方案，方案输出在附件。2023 年 1 月关停$DC9$后包裹都能正常流转，新增 3 条运输线路，变化 57 条优化原有线路。

七、问题 4 的模型建立与求解

7.1 问题 4 的分析

问题 4 第一部分是一个评价问题，第二部分是一个优化问题并需要讨论其鲁棒性。本问题的难点在于此问题并没有给出衡量物流场地和运输线路重要性的指标，也没有明确增加场地和线路的数量更没有给出衡量新增场地和新增线路的表述，是一道开放的问题。如果全面考虑原问题，会使模型极为复杂，因此考虑合理假设简化问题。

对于第一部分，本文首先根据查阅资料人为选取衡量场地和运输线路重要性的指标，再利用主成分分析法给出评价的判别表达式和结果。

对于第二部分，首先明确新增物流场地和线路的目的，即缓解已有场地和线路的货量压力，同时还要保证新增物流场地和运输线路的货量压力不会太大。因此需要建设确定建设成本和建设效益的模型，构建优化模型确定新增场地的运输线路选择方案，最终采用优化算法求解。对问题 1 预测结果加以随机扰动，观察因此改变的新增场地和新增线路数量，分析所构建网络的鲁棒性。

问题 4 第一部分流程图如图7.1。

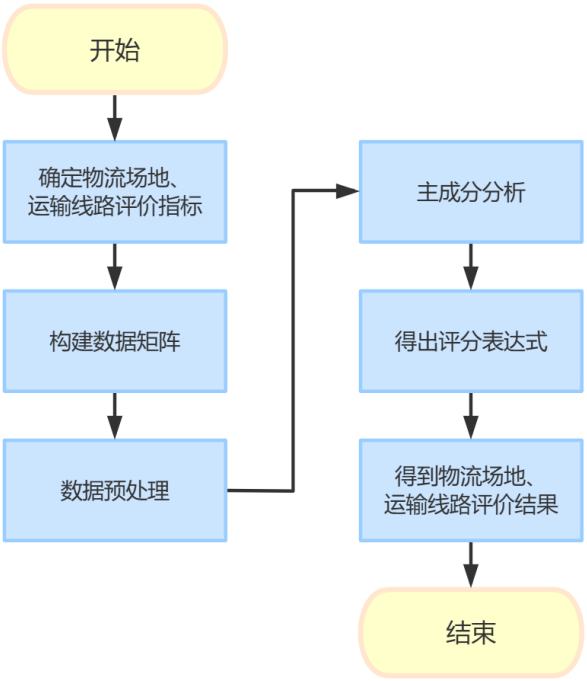

图 7.1　问题 4 第一部分流程图

问题 4 第二部分流程图如图 7.2。

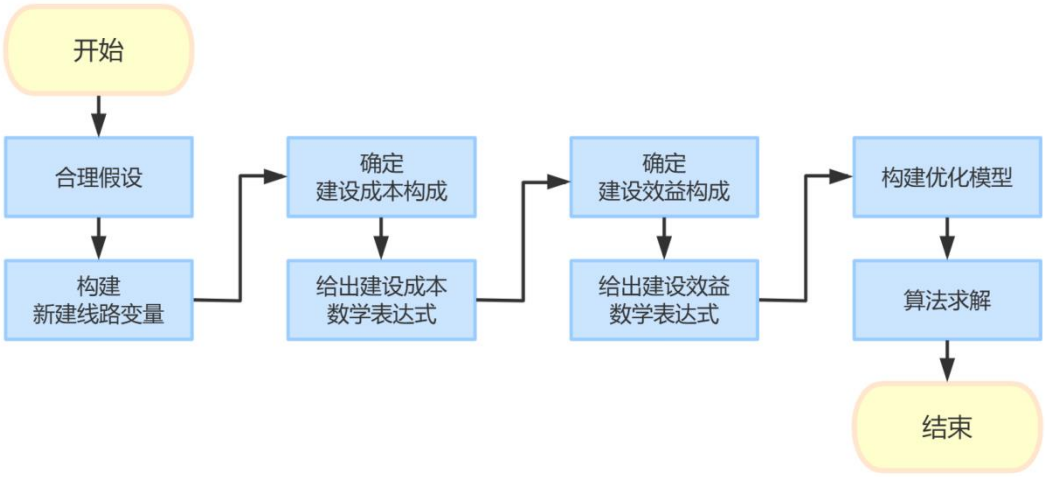

图 7.2　问题 4 第二部分流程图

7.2　第一部分——重要性评价模型的建立

为选取评价网络不同物流场地及线路的重要性，需要选取合理的指标，经查阅文献[3][5]后选择以下 12 个指标来衡量物流场地的重要性：历史最大收货量、历史最小收货量、历史平均收货量、历史最大发货量、历史最小发货量、历史平均发货量、收货总天数、发货总天数、同时收发货天数、最大邻接点数、最小邻接点数、平均邻接点数。

查阅相关文献后选取以下 8 个指标来衡量运输线路的重要性：历史最大货量、历史最小货量、历史平均货量、货量存在天数、最长货量存在天数、最长货量不存在天数、

该边货量占左相连场地的总发货线路数比例、该边货量占右相连场地的总收货线路数比例。

可以观察到所选取的指标有一部分指标存在共线性问题，因此采用主成分分析法建立重要性评价模型。

7.2.1 数据标准化处理

设物流场地的衡量指标为：$a_{\cdot j}$，$j=1,2,\cdots,12$。设运输线路的衡量指标为：$b_{\cdot j}$，$j=1,2,\cdots,8$。记一共有n个物流场地，m条运输线路。

对物流场地的数据进行标准化处理：

$$a'_{ij}=\frac{a_{ij}-u_j}{s_j}，\ i=1,2,\cdots,n;j=1,2,\cdots,12 \tag{7.1}$$

其中

$$u_j=\frac{\sum_{i=1}^{n}a_{ij}}{n}，\ s_j=\sqrt{\frac{\sum_{i=1}^{n}(a_{ij}-u_j)^2}{n-1}}$$

对运输线路的数据进行标准化处理：

$$b'_{ij}=\frac{b_{ij}-u'_j}{s'_j}，\ i=1,2,\cdots,m;j=1,2,\cdots,8 \tag{7.2}$$

其中

$$u'_j=\frac{\sum_{i=1}^{n}b_{ij}}{m}，\ s'_j=\sqrt{\frac{\sum_{i=1}^{n}(b_{ij}-u'_j)^2}{n-1}}$$

7.2.2 计算两组数据的相关系数矩阵

物流场地相关系数矩阵R_1，R_1中的元素r_{ij}^1为：

$$r_{ij}^1=\frac{\sum_{k=1}^{n}a'_{ki}a'_{jk}}{n-1}，\ i,j=1,2,\cdots,12 \tag{7.3}$$

运输线路相关系数矩阵R_2，R_2中的元素r_{ij}^2为：

$$r_{ij}^2=\frac{\sum_{k=1}^{n}b'_{ki}b'_{jk}}{m-1}，\ i,j=1,2,\cdots,8 \tag{7.4}$$

7.2.3 计算特征值和特征向量

计算相关系数矩阵R_1、R_2的特征值$\lambda_1^1\geq\lambda_2^1\geq\cdots\geq\lambda_{12}^1\geq0$，$\lambda_1^2\geq\lambda_2^2\geq\cdots\geq\lambda_8^2\geq0$以及对应的标准化后的特征向量$u_1^1,u_2^1,\cdots,u_{12}^1$和$u_1^2,u_2^2,\cdots,u_8^2$。$R_1$的每一个特征向量为 12 维，$R_2$的每一个特征向量为 8 维。

那么由R_1的特征向量组成的 12 个新指标为：$y_1^1=(u_1^1)^Ta,y_2^1=(u_2^1)^Ta,\cdots,y_{12}^1=(u_{12}^1)^Ta$，其中$a$为 11 个指标组成的向量。由$R_2$的特征向量组成的 8 个新指标为：$y_1^2=(u_1^2)^Tb,y_2^2=(u_2^2)^Tb,\cdots,y_8^2=(u_8^2)^Tb$，其中$b$为 8 个指标组成的向量。

7.2.4 选取主成分

分别选择p_1、p_2个主成分，其中p_1、p_2满足：

$$\frac{\sum_{k=1}^{p_1}\lambda_k^1}{\sum_{i=1}^{12}\lambda_i^1}>0.85 \quad \wedge \quad \frac{\sum_{k=1}^{p_1-1}\lambda_k^1}{\sum_{i=1}^{12}\lambda_i^1}<0.85 \tag{7.5}$$

$$\frac{\sum_{k=1}^{p_2}\lambda_k^2}{\sum_{i=1}^{8}\lambda_i^2}>0.85 \quad \wedge \quad \frac{\sum_{k=1}^{p_2-1}\lambda_k^2}{\sum_{i=1}^{8}\lambda_i^2}<0.85 \tag{7.6}$$

7.2.5 计算得分

给出各组数据得分表达式：

$$Z_1=\sum_{j=1}^{p_1}b_j^1y_j^1\text{，其中}b_j^1=\frac{\lambda_j^1}{\sum_{i=1}^{11}\lambda_i^1} \tag{7.7}$$

$$Z_2=\sum_{j=1}^{p_2}b_j^2y_j^2\text{，其中}b_j^2=\frac{\lambda_j^2}{\sum_{i=1}^{8}\lambda_i^2} \tag{7.8}$$

由此得到主成分评价函数，计算每个物流场地和运输线路的得分，评价不同场地和线路的重要性。

7.3 第二部分——最优新增决策优化模型的建立

7.3.1 模型合理假设

建立模型之前，做出合理性假设，本题应用到的假设在“二、模型假设”部分假设三到假设八。

7.3.2 构建新增路线变量

在本问题模型的合理假设下，只需要考虑新增一个物流场地的最优方案即可。因为由模型假设可知，如果新增多个场地，则将之前新增的场地和运输线路视为已有场地和线路，反复利用此模型即可求出最优方案。

在新增场地数量为 1 的情况下，本文首先构建新增线路变量，将新增场地视作第 0 个场地$DC0$。定义如下变量：

x_{0j}：以新增场地$DC0$为起点，DCj为终点的运输线路的运货量。若没有新增线路，则$x_{0j}=0$；若有新增线路，则$x_{0j}>0$。

x_{i0}：以DCi为起点，新增场地$DC0$为终点的运输线路的运货量。若没有新增线路，则$x_{i0}=0$；若有新增线路，则$x_{i0}>0$。

由假设可知新增线路每天的运货量不变，所以将 2023-01-01 至 2023-01-31 期间新增线路的运货量均为此数量。

查阅相关文献可知，新增线路应使建设成本达到最小，建设效益达到最大，下面分别构建有关建设成本和建设效益的目标函数。

7.3.3 关于建设成本的目标函数

一、构建建设成本的目标函数

查阅相关文献[4]知，新增一个物流场地和若干运输线路由以下三部分构成：①建设每一条线路的距离成本；②线路运货成本；③物流场地的收发货成本。基于这三部分，给出各物流场地建设成本的表达式：

由假设知每条线路的距离大致相同，因此建设每条线路的距离成本为：

$$C_1 = c_1 \sum_{i=1}^{81} sgn(x_{i0}) + c_1 \sum_{j=1}^{81} sgn(x_{0j}) \tag{7.9}$$

其中c_1为建设一条线路的成本。

由假设知新增线路的工作负荷量为现有线路负荷量的平均值，记为η。而运货量成本由线路运货量上限决定。因此线路运货量成本为：

$$C_2 = c_2 \frac{\sum_{i=1}^{n} x_{i0}}{\eta} + c_2 \frac{\sum_{j=1}^{n} x_{0j}}{\eta} \tag{7.10}$$

其中c_2为单条线路运货的成本。

由假设知新增场地的处理能力取决于新增运输线路的数量和每条运输线路的最大运货量。给出新增场地发货量的最大值为$\sum_{j=1}^{n} x_{0j}/\eta$和收货量的最大值为$\sum_{i=1}^{n} x_{i0}/\eta$。定义单个场地的处理成本为$c_3$，新增场地货量的处理成本为：

$$C_3 = c_3 \left(\frac{\sum_{j=1}^{n} x_{0j}}{\eta} + \frac{\sum_{i=1}^{n} x_{i0}}{\eta} \right) \tag{7.11}$$

因此整理并化简式(???)到式(???)可以构建关于成本的目标函数为：

$$\begin{aligned} C &= C_1 + C_2 + C_3 \\ &= c_1 \left[\sum_{i=1}^{81} sgn(x_{i0}) + \sum_{j=1}^{81} sgn(x_{0j}) \right] + \frac{(c_2 + c_3)\left(\sum_{j=1}^{n} x_{0j} + \sum_{i=1}^{n} x_{i0}\right)}{\eta} \end{aligned} \tag{7.12}$$

7.3.4 关于建设效益的目标函数

（一）基于本题假设 6，定义新增后的集合：

$E_{10}\{j|x_{0j} \neq 0\}$：与新增场地右相连的场地（发货线路相连的场地）

$E_{20}\{i|x_{i0} \neq 0\}$：与新增场地左相连的场地（收货线路相连的场地）

对$\forall j \in E_{10}, i \in E_{20}$，定义已有发货线路集合为$E_{1j}\{(k,j)|x_{kj} \neq 0\}$，已有收货线路集合$E_{2i}\{(i,m)|x_{im} \neq 0\}$

（二）计算$\forall j \in E_{10}, i \in E_{20}$，集合$E_{1j}$，$E_{2i}$中每一条线路应减小的百分比与新增运货线路后的流量：

对于$\forall j \in E_{10}$，线路运货量减小百分比为：

$$k = \frac{x_{0j}}{\sum_{(k,j)\in E_{1j}} x_{kj}} \tag{7.13}$$

因此对$\forall (k,j) \in E_{1j}$，新增运货线路后的线路流量为：$x'_{kj} = x_{kj}(1-k)$。

同理，对于$\forall i \in E_{20}$，线路运货量减小百分比为：

$$k' = \frac{x_{i0}}{\sum_{(i,m)\in E_{2i}} x_{im}} \tag{7.14}$$

因此对于$\forall (i,m) \in E_{2i}$，新增运货线路后的线路流量为$x'_{im} = x_{im}(1-k')$。

（三）构建新增场地和线路的建设效益数学表达式：

问题 4 第一部分得出的线路评分表达式：

$$Z_2 = \sum_{j=1}^{p_2} b_j^2 y_j^2 \text{，其中} b_j^2 = \frac{\lambda_j^2}{\sum_{i=1}^{8} \lambda_i^2}$$

定义未新增场地和线路之前，基于问题 1 预测结果的线路(i,j)的第n个主成分大小为$B_n^{(i,j)}$，其中$0 < n < 9$。

新增场地和线路之后，问题 1 预测的结果需要进行处理以满足假设条件，即利用公式把问题 1 预测的结果转化为新增运货线路后的线路流量。定义线路(i,j)减小运货量之后的第n个主成分大小为$B_n^{(i,j)'}$，那么对于每一条线路(i,j),都有新增场地和线路之前的得分，将其分别定义为：

$$Z_{(i,j)} = \sum_{i=1}^{p_2} b_i B_i^{(i,j)} \tag{7.15}$$

$$Z'_{(i,j)} = \sum_{i=1}^{p_2} b_i B_i^{(i,j)'} \tag{7.16}$$

本文将所有改变运货量线路的得分减小值综合记为建设效益，其实际意义为：得分变小时，此线路工作负荷量变小，而新增线路的意义就是为了减小已有存在线路的工作工作负荷。因此可以得到关于建设效益的目标函数为：

$$A = \sum_{j\in E_{10}} \sum_{(k,j)\in E_{1j}} \left(Z_{(k,j)} - Z'_{(k,j)}\right) - \sum_{i\in E_{20}} \sum_{(i,m)\in E_{2i}} \left(Z_{(i,m)} - Z'_{(i,m)}\right) \tag{7.17}$$

7.3.5 构建约束条件

首先新建线路不能过多，规定新增线路不超过 15 条：

$$\sum_{k=1}^{81} sgn(x_{0k}) + \sum_{k=1}^{81} sgn(x_{k0}) \le 15 \tag{7.18}$$

新增线路的工作负荷不能超过所有线路历史最大负荷量：

$$x_{0k}/\eta\,, x_{k0}/\eta \le M,\ \forall k \in (1,2,\cdots,81) \tag{7.19}$$

7.3.6 多目标转化为单目标并构建优化模型

上述两个目标函数可以转化为单目标优化模型，优化模型为：

$$\max \frac{\sum_{j\in E_{10}}\sum_{(k,j)\in E_{1j}}\left(Z_{(k,j)}-Z'_{(k,j)}\right)+\sum_{i\in E_{20}}\sum_{(i,m)\in E_{2i}}\left(Z_{(i,m)}-Z'_{(i,m)}\right)}{c_1\left[\sum_{i=1}^{81}sgn(x_{i0})+\sum_{j=1}^{81}sgn\left(x_{0j}\right)\right]+(c_2+c_3)(\sum_{j=1}^{n}x_{0j}+\sum_{i=1}^{n}x_{i0})/\eta}$$

$$\text{s.t.}\begin{cases} k=\dfrac{x_{0j}}{\sum_{(k,j)\in E_{1j}}x_{kj}},\ x'_{kj}=x_{kj}(1-k) \\ k'=\dfrac{x_{i0}}{\sum_{(i,m)\in E_{2i}}x_{im}},\ x'_{im}=x_{im}(1-k') \\ Z_{(i,j)}=\sum_{i=1}^{p_2}b_iB_i^{(i,j)},\ Z'_{(i,j)}=\sum_{i=1}^{p_2}b_iB_i^{(i,j)'} \\ \sum_{k=1}^{81}sgn(x_{0k})+\sum_{k=1}^{81}sgn(x_{k0})\le 15 \\ x_{0k}/\eta\,, x_{k0}/\eta \le M,\ \forall k \in (1,2,\cdots,81) \end{cases} \tag{7.20}$$

7.4 问题 4 的模型求解

7.4.1 重要性评价模型的求解

一、评价物流场地重要性

首先观察标准化后相关系数矩阵的数据，发现物流场地相关系数矩阵R_1中历史最小发货量、历史最小收货量、最小邻接点数三个指标的所有数据均为 0，因此首先删除这 3 个指标，留下 9 个指标进行主成分分析。

利用$SPSS$软件对这 9 个指标进行主成分分析，得到相关系数矩阵R_1的特征值。将特征值从大到小排序并计算其信息贡献率和累计贡献率，结果如表 7.1 所示。

表 7.1 物流场地重要性主成分分析计算结果

顺序	特征值	信息贡献率（%）	累计贡献率（%）
1	5.930	65.887	65.887
2	1.393	15.480	81.368
3	1.180	13.116	94.484
4	0.364	4.043	98.527
5	0.101	1.119	99.646
6	0.025	0.280	99.925
7	0.004	0.041	99.966
8	0.002	0.022	99.988
9	0.001	0.012	100.000

以累计贡献率≥ 85%选取主成分，选择特征值从大到小的前 3 个指标分别作为第一主成分、第二主成分、第三主成分。计算成分得分系数矩阵，计算结果如表 7.2。

表 7.2　成分得分系数矩阵计算结果

指标x_i	第一主成分	第二主成分	第三主成分
历史最大收货量	-0.154	0.197	0.182
历史平均收货量	-0.155	0.201	0.153
历史最大发货量	-0.123	0.021	-0.571
历史平均发货量	-0.125	0.042	-0.561
收货总天数	-0.133	0.065	0.337
发货总天数	-0.090	-0.571	-0.008
同时收发货天数	-0.100	-0.537	0.136
最大邻接点数	-0.166	0.046	0.098
平均邻接点数	-0.163	0.122	0.091

由此得到三个主成分的表达式：

$$Y_1 = -0.154x_1 - 0.155x_2 - 0.123x_3 - 0.125x_4 - 0.133x_5 - 0.09x_6 - 0.1x_7 - 0.166x_8 - 0.163x_9$$

$$Y_2 = 0.197x_1 + 0.201x_2 + 0.021x_3 + 0.042x_4 + 0.065x_5 - 0.571x_6 - 0.537x_7 + 0.046x_8 + 0.122x_9$$

$$Y_3 = 0.182x_1 + 0.153x_2 - 0.571x_3 - 0.561x_4 + 0.337x_5 - 0.008x_6 + 0.136x_7 - 0.098x_8 - 0.091x_9$$

评价物流场地的主成分得分函数为：

$$Z_1 = 0.6588Y_1 + 0.1548Y_2 + 0.1311Y_3$$

代入数据得第一、二、三重要的物流场地分别为$DC10$、$DC14$、$DC5$；物流场地重要性最差的三个物流场地分别为$DC74$、$DC75$、$DC77$。其余所有数据均在附件中。

二、评价运输线路重要性

观察发现运输线路标准化后的相关系数矩阵R_2中历史最小货量指标数据均为 0，因此首先删除这个指标，留下 7 个指标进行主成分分析。

利用$SPSS$软件对这 7 个指标进行主成分分析，得到相关系数矩阵R_2的特征值。将特征值从大到小排序并计算其信息贡献率和累计贡献率，结果如表 7.3 所示。

表 7.3　运输线路重要性主成分分析计算结果

顺序	特征值	信息贡献率（%）	累计贡献率（%）
1	3.344	47.775	47.775
2	1.346	19.230	67.006
3	0.993	14.185	81.191
4	0.737	10.526	91.716
5	0.380	5.435	97.151
6	0.113	1.616	98.767
7	0.086	1.233	100.000

以累计贡献率≥ 85%选取主成分，选择特征值从大到小的前 3 个指标分别作为第一主成分、第二主成分、第三主成分。计算成分得分系数矩阵，计算结果如表 7.4。

表 7.4　成分得分系数矩阵计算结果

指标x_i	第一主成分	第二主成分	第三主成分
历史最大货量	-0.225	-0.409	-0.064
历史平均货量	-0.219	-0.432	-0.077
货量存在天数	-0.257	0.216	0.119
最长货量存在天数	0.217	-0.319	-0.244
最长货量不存在天数	-0.261	0.280	0.161
该边货量占左相连场地的总发货线路数比例	-0.138	-0.120	-0.584
该边货量占右相连场地的总发货线路数比例	-0.019	0.381	-0.745

由此得到三个主成分的表达式：

$$W_1 = -0.225x_1 - 0.219x_2 - 0.257x_3 + 0.217x_4 - 0.261x_5 - 0.138x_6 - 0.019x_7$$

$$W_2 = -0.409x_1 - 0.432x_2 + 0.216x_3 - 0.319x_4 + 0.280x_5 - 0.120x_6 + 0.381x_7$$

$$W_3 = -0.064x_1 - 0.077x_2 + 0.119x_3 - 0.244x_4 + 0.161x_5 - 0.584x_6 - 0.745x_7$$

评价物流场地的主成分得分函数为：

$$Z_2 = 0.6588W_1 + 0.1548W_2 + 0.1311W_3$$

代入数据得第一、二、三重要的运输线路分别为$DC58 \to DC29$、$DC29 \to DC9$、$DC29 \to DC36$；运输线路重要性最差的三个运输线路分别为$DC51 \to DC9$、$DC14 \to DC9$、$DC14 \to DC8$。其余所有数据均在附件中。

7.4.2　最优新增决策优化模型的求解

具体举例一个结果，以 2023 年第一周（2023-01-01 至 2023-01-07）为例,流入新增场地的场地为：$DC56$、$DC1$、$DC36$、$DC53$、$DC60$、$DC77$；流出新增场地的场地为$DC7$、$DC10$、$DC17$、$DC23$、$DC42$、$DC43$、$DC68$，流量为 2999。

具体流量分配方案见附件。

7.4.3　网络鲁棒性分析

图 7.3 为新增场地鲁棒性检验结果，对问题 1 预测结果（即 2023 年 1 月预测结果）加以 0 到 220%的随机扰动，分析问题 4 构建的网络的鲁棒性。由图 7.3 可知，当允许调度方案退化 20%时，允许网络中货量扰动不超过 130%，可以应对大部分临时变化，但仍然无法应对“618”,“双十一”等重要电商促销活动，因为由网络总流量的历史数据与预测结果得知，电商促销的峰值是现有运行网络流量的 220%。历史上出现了大量大容量“临时线路”应对电商节的多余流量，这样的方案不被我们的调度优化模型接受。因此可以得出此网络具有较强的鲁棒性。

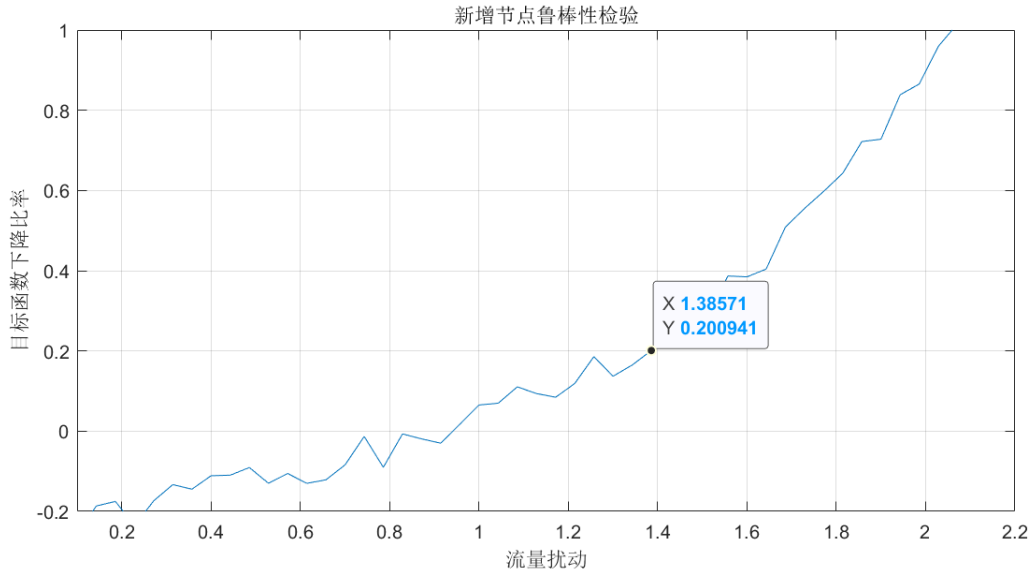

图 7.3　网络鲁棒性分析结果图

八、模型评价

8.1　模型的优点

1. 问题 1 模型的数据预处理步骤较多但考虑的十分全面，可以很好的处理因为疫情和购物节等因素使得的货流量长时间为异常值的路径数据，处理之后的数据可以视为没有特殊因素影响的数据。
2. 问题 1 模型的预测方法回归阶数较大，考虑到了数据具有周期性，而且对数据进行了去噪，得出的预测结果更使人信服。
3. 问题 2、问题 3 的模型考虑的较为全面，可以很好的刻画出因关停$DC5$、$DC9$场地在假设情况下可以分配的所有可能情况。将原始的复杂优化模型化简，得到较为简洁的优化模型，方便计算机求解。
4. 问题 2、问题 3 求解过程中，采用了外部遗传算法、内部模拟退火算法相组合的求解方法，相较于单一的优化算法更加高效，快速。同时给出了一个自创的产生随机可行解的方法，求解过程中采用了并行计算，加快求解速度。
5. 问题 4 的评价模型里，查阅相关资料后选取尽可能多的指标进行评价，保证了评价的全面性。采用主成分评价法也可以使最后的评价函数里消除评价指标产生的共线性问题。
6. 针对问题 4 的一些具有合理性假设可以很好降低此问题的复杂程度，使得建立的模型不会过于复杂，方便计算机求解。
7. 问题 4 对建立的模型进行简化，并给出效率更高的计算方法，加快求解效率。

8.2 模型的缺点

1. 由于各种因素的影响，数据过于复杂，即使对数据进行了较多的处理，利用验证集验证时平均误差 10%左右，无法使得预测结果非常可信，泛化性还有待提高。
2. 问题 2、问题 3、问题 4 模型中使用较多的集合表示节点和路径的范畴，虽然简化了模型的表述，但容易混淆。计算机求解时也需要遍历很多次原始数据，当数据量十分庞大（例如几万个节点，几十万条边）时，适用性较差。
3. 问题 2、问题 3、问题 4 虽然模型的约束条件即为题目要求，但并没有找到进一步简化此约束条件的办法，因此模型中约束条件过多，算法求解时的可行解的产生需要一定的时间。
4. 在考虑新建多个场地时假设把建设方式设定为新建完一个节点再新建下一个，没有考虑同时建设的方案。

8.3 模型的改进

1. 问题 1 模型的数据预处理方法有待改进，可以考虑利用傅里叶变换等方法去除异常值。
2. 问题 2 求解时利用外层遗传算法、内层模拟退火算法求解，$Matlab$需要运行很长时间才能跑出结果。观察到遗传和模拟退火可以进一步改进为量子退火模型（如$QUBO$模型）进而加快运算速度。
3. 问题 2、3 中模型假设较少、变量较多，可以考虑增加合理性假设以简化模型。
4. 问题 4 第一部分评价模型采用基于主成分分析法的评价模型，主成分分析降维后，存在少量信息丢失；标准化处理之后，含义会发生变化，解释性含义时具有模糊性。可以考虑基于熵权法或改进的主成分分析法的评价模型。
5. 问题 4 第二部分假设虽然是合理的，但假设过多，且最终建立的模型仍较为复杂。可以考虑留下重要的假设并增加重要合理的假设，以简化模型。

九、参考文献

[1]解婧瑶,童慧,陈立中.基于层次分析法和主成分分析法的仓库 KPI 评价体系构建——以中国外运公司为例[J].价值工程,2022,41(23):67-69.

[2] 胡佳迎.关于快递行业中货量预测方法的介绍[J].电脑知识与技术,2018,14(08):152-153.

[3]李朝迁,裴建朝.新型模拟退火遗传算法在路径优化的应用[J].组合机床与自动化加工技术,2022,No.577(03):52-55.

[4]石褚巍,马昌喜,麻存瑞.基于两阶段鲁棒优化的可靠性物流网络设计[J/OL].交通运输系统工程与信息:1-20[2023-04-16].

[5]林壮钦. 基于鲁棒优化方法的不确定环境下建筑垃圾中转站选址研究[D].重庆大学,2020.

附录

附录一：构建邻接矩阵

CreateGragh.m

```
clc;clear;
load originGragh.mat
%% 初始化
[n,m]=size(Gragh);
tmp_gragh=zeros(n,m);
for i=1:n
    tmp_gragh(i,1)=str2num(erase(Gragh(i,1),"DC"));
    tmp_gragh(i,2)=str2num(erase(Gragh(i,2),"DC"));
end
Gragh=tmp_gragh;
clear i tmp_gragh
Nodes=unique(Gragh);
num=length(Nodes);
filename = 'Gragh.xlsx';

%% 计算连通性
Matrix_Connect=zeros(num);
for i=1:num
    Matrix_Connect(i,i)=inf;
end
num_connect=0;
for i=1:n
    Matrix_Connect(Gragh(i,1),Gragh(i,2))=Matrix_Connect(Gragh(i,1),Gragh(i,2))+1;
    if (Matrix_Connect(Gragh(i,1),Gragh(i,2))==1)
        num_connect=num_connect+1;
    end
end
clear i
writematrix(Matrix_Connect,filename,'Sheet',1,'Range','B2:CD82');
writematrix(1:num,filename,'Sheet',1,'Range','B1:CD82');
writematrix([1:num]',filename,'Sheet',1,'Range','A2:A82');
%% 清理
clear n m ans filename
```

TotalFlow.m

```
%% 计算和
Total=zeros(length(ind_start)+1,365*2);
for i=1:n
    for j=1:length(ind_start)
        if (ind_start(j)==Gragh(i,1))
```

```
                dates=days(Date(i)-Date(1))+1;
                Total(j,dates)=Total(j,dates)+Volume(i);
            end
        end
end
Total(length(ind_start)+1,:)=sum(Total(1:length(ind_start),:));
%for i=1:365*2
%       Total(,i)=sum(Total(1:ind_start,i));
%end
clear i j
```

附录二：构建时间序列并预测

OriginSeries.m

```
load Gragh_Connect.mat
n=length(Gragh);
Time_var=zeros(num_connect,365*2);
Time_start=zeros(num_connect,1);
Time_end=zeros(num_connect,1);
Time_total=zeros(num_connect,1);
ind =0;
Calced=zeros(num);
for i=1:n
    if (Matrix_Connect(Gragh(i,1),Gragh(i,2))>0)
        if (Calced(Gragh(i,1),Gragh(i,2))==0)
            ind=ind+1;
            Calced(Gragh(i,1),Gragh(i,2))=ind;
            Time_start(ind)=Gragh(i,1);
            Time_end(ind)=Gragh(i,2);
        end
    end
end
clear i ind j
for i=1:n
    if (Matrix_Connect(Gragh(i,1),Gragh(i,2))>0)
        dates=days(Date(i)-Date(1))+1;
        ind=Calced(Gragh(i,1),Gragh(i,2));
        Time_var(ind,dates)=Volume(i);
        Time_total(ind)=Time_total(ind)+Volume(i);
    end
end
Index=[1:num_connect]';
tbl=table(Index,Time_total,Time_start,Time_end,Time_var);
tbl = sortrows(tbl,2,'descend');
writetable(tbl,'series.xlsx');
```

```
clear i dates ind Index
clear ans n Calced
clear tbl
```

cate_series.m

```
clc;clear
load Series.mat
%未按照流量排序
Count=cell(1,7);%七类
% 1 优秀节点
% 2 临时节点
% 3 延迟开启节点
% 4 永久关闭节点
% 5 短时间节点
% 6 式微节点
% 7 繁荣节点
crisis=120;
time=size(Time_var,2);
for i=1:num_connect
    IsTemp(i)=(Time_total(i)<=10) || ((nnz(Time_var(i,:))<=10));
    if (IsTemp(i)==1)
        Count{2}(end+1)=i;
    end
end
clear i
for i=1:num_connect
    if (IsTemp(i)==1)
        continue
    end
    Delay=1;%是否开始很长一段时间未运行
    Shut=1;%是否永久关闭
    for j=1:time
        if (j<crisis && Time_var(i,j)>0)
            Delay=0;
        end
        if (j>time-crisis && Time_var(i,j)>0)
            Shut=0;
        end
    end
    if (Delay+Shut==2)
        Count{5}(end+1)=i;
    end
    if (Delay==1)
        Count{3}(end+1)=i;%记录下节点编号
    end
```

```
    if (Shut==1)
        Count{4}(end+1)=i;%记录下节点编号
    end
    if (Delay+Shut==0)%既不是很长时间未运行也不是永久关闭
        Count{1}(end+1)=i;
    end
    clear Delay Shut Temp
end
clear i j
%对正常节点再细分
tmp=[];
for i=Count{1}
    delta=mean(Time_var(i,end-crisis+1:end))/mean(Time_var(i,crisis+1:end-crisis+1));
    if (log(delta)<-0.1 || max(Time_var(i,end-crisis+1:end))<10)
        Count{6}(end+1)=i;
    else
        if (log(delta)>1)
            Count{7}(end+1)=i;
        else
            tmp(end+1)=i;
        end
    end
end
Count{1}=tmp;
clear i tmp delta
save Cate.mat Count
    clear_series.m
clear ans time crisis IsTemp
clc;clear;close all;
load Cate.mat
load Series.mat
ind=[Count{1},Count{3},Count{6},Count{7}];
ind=sort(ind);
result=cell(1,3);
p=0;
%%
bias=10;%极小值的剔除偏置
for len=1:length(ind)
    tmp_Series_var{len}=Time_var(ind(len),:);
end
edges=0;
ind_edge=[];
for i=1:len
    %{
```

```
tmp_ind=[];
for j=1:365*2
    if (Series_var{i}(j)>min)
        tmp_ind=[tmp_ind,j];
    end
end
%}
tmp_ind=find(tmp_Series_var{i}>3);%下标
tmp_Series_var{i}=tmp_Series_var{i}(tmp_ind);

c=-1/(sqrt(2)*erfcinv(3/2));
Med=median(tmp_Series_var{i});
tmp=[];
MAD=2*c*median(abs(tmp_Series_var{i}-Med));
%{
if (i==145)
    disp(MAD);
    disp(Med);
    %system("pause");
end
%}
for j=1:length(tmp_Series_var{i})
    if (tmp_Series_var{i}(j)<=Med+MAD && tmp_Series_var{i}(j)>=Med-MAD)
        tmp=[tmp,tmp_Series_var{i}(j)];
    end
end
%tmp = rmoutliers(Series_var{i},'gesd');
clear c MAD j Med
%{
if (i==145)
    %disp(tmp);
    %disp(Series_var{i});e");
    setdiff(tmp_Series_var{i},tmp)
    %system("pause");
end
%}
tmp_ind=find(tmp>(max(tmp)*0.03));%下标
tmp=tmp(tmp_ind);
%tmp_ind=find(tmp>(Min+bias));%下标

%{
if (i==145)
    %disp(tmp);
    %disp(Series_var{i});
```

```
            setdiff(tmp_Series_var{i},tmp)
            %system("pause");
        end
        %}
        %{
        TF=isoutlier(tmp);
        ind_TF=find(TF);
        %tmp=filloutliers(tmp,'nearest');
        res_ind=0;res=[];
        for k=1:length(tmp)
            if nnz(ismember(ind_TF,k)>0)
                continue;
            end
            res_ind=res_ind+1;
            res=[res,tmp(k)];
        end
        Series_var{i}=res;
        %}

        if (length(tmp)>=5)
            edges=edges+1;
            Series_var{edges} = tmp;
            ind_edge(edges)=ind(i);
        end
        clear tmp_ind Min tmp
        clear TF ind_TF k res res_ind
end
clear i j bias
%% 去噪
for i=1:edges
        if (ind_edge(i)==33 || ind_edge(i)==61 || ind_edge(i)==729)
            p=p+1;
            result{p}=Series_var{i};
        end
        for j=3:length(Series_var{i})-2
            Series_var{i}(j)=mean(Series_var{i}(j-2:j+2));
        end
        Series_var{i}(1)=mean(Series_var{i}(1:1+2));
        Series_var{i}(2)=mean(Series_var{i}(1:2+2));
        Series_var{i}(end)=mean(Series_var{i}(end-2:end));
        Series_var{i}(end-1)=mean(Series_var{i}(end-3:end));

end
clear i
```

```
clear i j p
save("cleared_Series.mat","Series_var","ind_edge");
save("Problem1.mat","result");
clear ans

function str=Node(node)
    str=strcat("DC",num2str(node));
end
```

Daniel.m

```
clc;clear;close all;
load cleared_Series.mat
n=length(ind_edge);
p=0.95;
for i=1:n
    Data=Series_var{i};
    [sorted,ind_sorted]=sort(Data);
    len=length(Data);
    qs(i)=1-6/(len*(len^2-1))*sum(([1:len]-ind_sorted).^2);
    T(i)=qs(i)*sqrt(n-1)/sqrt(1-qs(i)^2);
    X(i)=tinv(p,len-2);
    clear sorted ind_sorted len Data
end
clear i n p
Flat=find(T<=X);
unFlat=find(T>X);
clear T X qs ans Series_var ind
save("DanielResult.mat");
%{
load cleared_Series.mat
n=length(ind);
p=0.95;
for i=unFlat
    Data=Series_var{i};
    [sorted,ind_sorted]=sort(Data);
    len=length(Data);
    qs(i)=1-6/(len*(len^2-1))*sum(([1:len]-ind_sorted).^2);
    T(i)=qs(i)*sqrt(n-1)/sqrt(1-qs(i)^2);
    X(i)=tinv(p,len-2);
    clear sorted ind_sorted len Data
end
%}
```

tranquil_series.m

```
clc;clear;close all;
%delete(gcp('nocreate'));
%p=parpool(12);
load cleared_Series.mat
load DanielResult.mat
%load dd_cleared.mat
len_block=40;%滑块长度
len_proving=10;%检验长度
len_grant=len_block+len_proving; %数据完整点判定阈值
len_pre=31;%预测长度
Predict=zeros(length(ind_edge),len_pre);
broken=0;
gap=0;
%total=1050;
%err=zeros(1,total);
%Predict=zeros(1,total);
%% 对于平稳序列
for ind=1:length(Flat)
    i=Flat(ind);
    ind_node=ind_edge(i);
    Data=Series_var{i};
    %Mean=Data-mean(Data);
    if (length(Data)<=5)
        gap=gap+1;
        continue
    end
    % 完整点
    if (length(Data)>=len_grant)
        len_block=40;%滑块长度
        len_proving=10;%检验长度

        len=length(Data);
        tmp_data=Data(1:len-len_proving);

        len=len-len_proving;
        for j=1:len_proving
            len=len+1;
                tmp_data(len)=Average(tmp_data,len_block);

        end

        err(ind_node)=abs(mean((tmp_data(len-len_proving:len)-Data(len-len_proving:len))./Data(len-
len_proving:len)));
        tmp_data=Data;
```

```
        for j=1:len_pre
            len=len+1;
            try
            model = arima(5,0,1);
            fit = estimate(model,tmp_data(len-len_block:len-1)');
            %fit = estimate(model,tmp_data');
            fore = forecast(fit,1);
            tmp_data(len)=fore;
            %[a,b]=Smoothing(tmp_data,0.3);
            %tmp_data(len)=a+b;
            catch
                [a,b]=Smoothing(tmp_data,0.3);
                tmp_data(len)=a+b;
            end

        end
        Predict(ind_node,:)=tmp_data(len-len_pre+1:len);
        %clear len a b

else
    % 非完整点
        len_block=2;%滑块长度
        len_proving=2;%检验长度

        len=length(Data);
        tmp_data=Data(1:len-len_proving);
        len=len-len_proving;
        for j=1:len_proving
            len=len+1;
            %[a,b]=Smoothing(tmp_data,0.3);
            %tmp_data(len)=a+b;
            tmp_data(len)=Average(Data,len_block);
        end
        err(ind_node)=abs(mean((tmp_data(len-len_proving:len)-Data(len-len_proving:len))./Data(len-
len_proving:len)));
        for j=1:len_pre
            [a,b]=Smoothing(tmp_data,0.3);
            len=len+1;
            tmp_data(len)=a+b;
        end
        Predict(ind_node,:)=tmp_data(len-len_pre+1:len);
        broken=broken+1;
        %clear len a b
```

```
    end
end
clear ind i j ans Data

%% 对于非平稳序列
for ind=1:length(unFlat)
    i=unFlat(ind);
    ind_node=ind_edge(i);
    Data=diff(Series_var{i});
    %Mean=Data-mean(Data);
    if (length(Data)<=5)
        gap=gap+1;
        continue
    end
    % 完整点
    if (length(Data)>=len_grant)
        len_block=40;%滑块长度
        len_proving=10;%检验长度

        len=length(Data);
        tmp_data=Data(1:len-len_proving);
        len=len-len_proving;
        for j=1:len_proving
                tmp_data(len)=Average(tmp_data,len_block);
        end
        err(ind_node)=abs(mean((tmp_data(len-len_proving:len)-Data(len-len_proving:len))./Data(len-
len_proving:len)));
        tmp_data=Data;
        for j=1:len_pre
            len=len+1;
            try
            model = arima(5,0,1);
            fit = estimate(model,tmp_data(len-len_block:len-1)');
            %fit = estimate(model,tmp_data');
            fore = forecast(fit,1);
            tmp_data(len)=fore;
            %[a,b]=Smoothing(tmp_data,0.3);
            %tmp_data(len)=a+b;
            catch
                [a,b]=Smoothing(tmp_data,0.3);
                tmp_data(len)=a+b;
            end
        end
        Predict(ind_node,:)=tmp_data(len)+tmp_data(len-len_pre+1:len);
```

```
            %clear len a b
        else

        %  非完整点
            len_block=2;%滑块长度
            len_proving=2;%检验长度

            len=length(Data);
            tmp_data=Data(1:len-len_proving);
            len=len-len_proving;
            for j=1:len_proving
                len=len+1;
                %[a,b]=Smoothing(tmp_data,0.3);
                %tmp_data(len)=a+b;
                tmp_data(len)=Average(Data,len_block);
            end
            err(i)=abs(mean((tmp_data(len-len_proving:len)-Data(len-len_proving:len))./Data(len-
len_proving:len)));
            for j=1:len_pre
                [a,b]=Smoothing(tmp_data,0.3);
                len=len+1;
                tmp_data(len)=a+b;
            end
            Predict(i,:)=tmp_data(len)+tmp_data(len-len_pre+1:len);
            broken=broken+1;
            %clear len a b

        end
end
clear ind i j ans Data
%%  结论输出
%err=reshape(err,33,19);
Predict=max(Predict,0);
err=err(~isnan(err));
err=err(~isinf(err));
writematrix(err','err.xlsx');
writematrix(Predict,'Predict.xlsx');
save predictResult.mat Predict err
delete(p);
%%  平滑函数
function [a_pre,b_pre]=Smoothing(y,alpha)
        n=length(y);
        st1=zeros(1,n);st2=zeros(1,n);
        st1(1)=y(1);                st2(1)=y(1);
```

```
    start=min(100,max(2,n-5));
    for i=start:n
        st1(i)=alpha*y(i)+(1-alpha)*st1(i-1);
        st2(i)=alpha*st1(i)+(1-alpha)*st2(i-1);
    end
    a_pre=2*st1(n)-st2(n);
    b_pre=alpha/(1-alpha)*(st1(n)-st2(n));
end

function y_pre=Average(y,n)
    m=length(y);
    y_pre=sum( y(m-n+1:m) ) / n;
end
```

predict1.m

```
clc;clear;close all;
load Problem1.mat
len_proving=10;%检验长度
len_pre=31;%预测长度
Predict=zeros(3,len_pre);
err=zeros(1,3);
figure(2);
path(1,1)=14;path(1,2)=10;
path(2,1)=20;path(2,2)=35;
path(3,1)=25;path(3,2)=62;
for i=1:3
    subplot(3,1,i);
    Data=result{i};
        len=length(Data);
        tmp_data=Data(1:len-len_proving);
        len=len-len_proving;
        for j=1:len_proving
            len_block=20;%滑块长度
            len=len+1;
            %model = arima(5,0,1);
            %fit = estimate(model,tmp_data(len-len_block:len-1)');
            %fit = estimate(model,tmp_data');
            %fore = forecast(fit,1);
            %tmp_data(len)=fore;
            %[a,b]=Smoothing(tmp_data,0.3);
            tmp_data(len)=Average(tmp_data,len_block);
            %tmp_data(len)=Adaptive_filtering(tmp_data,len_block);
        end
        %bias=randn(1,10)*std(result{i})*0.3;
```

```
        %tmp_data(len-len_proving+1:len)=tmp_data(len-len_proving+1:len)+bias;
        err(i)=abs(mean((tmp_data(len-len_proving:len)-Data(len-len_proving:len))./Data(len-
len_proving:len)));
        tmp_data=Data;
        for j=1:len_pre
            len_block=40;%滑块长度
            len=len+1;
            model = arima(5,0,1);
            fit = estimate(model,tmp_data(len-len_block:len-1)');
            %fit = estimate(model,tmp_data');
            fore = forecast(fit,1);
            tmp_data(len)=fore;
            %[a,b]=Smoothing(tmp_data,0.3);
            %tmp_data(len)=a+b;
        end
    hold on
        bias=randn(1,31)*std(result{i})*0.5+exp(1:0.1:4);bias(1)=bias(1)+5;
        Predict(i,:)=max(tmp_data(len-len_pre+1:len)+bias,0);
        plot([datetime(2022,11,12:30),datetime(2022,12,1:31)],Data(end-50+1:end),'b-');
        %plot(41:81,[tmp_data(end-len_proving+1:end),Predict(i,:)],'r+-');
        plot(datetime(2023,1,1:31),Predict(i,:),'r+-');
        xlim([datetime("2022-11-12") datetime("2023-01-31")]);
        ylabel("货量");
        legend("历史数据","预测数据");
        title(strcat(Node(path(i,1)),' to ',Node(path(i,2)),'2023 年 1 月预测结果'),strcat("相对误差为
",num2str(err(i)*100),"%"));
        %datetick('x','mm-dd');
    hold off
    clear len a b
end
writematrix(Predict,'goal_predict.xlsx','Range','B2:AF4');
writematrix([1:3]','goal_predict.xlsx','Range','A2:A4');
writematrix([1:31],'goal_predict.xlsx','Range','B1:AF1');
clear i j
function [a_pre,b_pre]=Smoothing(y,alpha)
    n=length(y);
    st1=zeros(1,n);st2=zeros(1,n);
    st1(1)=y(1);            st2(1)=y(1);
    start=min(100,max(2,n-5));
    for i=start:n
            st1(i)=alpha*y(i)+(1-alpha)*st1(i-1);
            st2(i)=alpha*st1(i)+(1-alpha)*st2(i-1);
    end
    a_pre=2*st1(n)-st2(n);
```

```
    b_pre=alpha/(1-alpha)*(st1(n)-st2(n));
end
function y_pre=Average(y,n)
    m=length(y);
    y_pre=sum( y(m-n+1:m) ) / n;
end
function str=Node(node)
    str=strcat("DC",num2str(node));
end
```

附录三：计算节点处理能力

maxVolume.m

```
clc;clear;close all;
load Series.mat
maxPop=zeros(1,num);
maxPush=zeros(1,num);
n=length(Gragh);
ind_start=1;
while (ind_start<=n)%按日起算
    ind_end=During(ind_start,Date,n);
    tmpPop=zeros(1,num);
    tmpPush=zeros(1,num);
    for edge=ind_start:ind_end
        tmpPop(Gragh(edge,1))=tmpPop(Gragh(edge,1))+Volume(edge);
        tmpPush(Gragh(edge,2))=tmpPush(Gragh(edge,2))+Volume(edge);
    end
    for i=1:num
        maxPop(i)=max(maxPop(i),tmpPop(i));
        maxPush(i)=max(maxPush(i),tmpPush(i));
    end
    ind_start=ind_end+1;
    clear i edge
end
clear ind_start ind_end ans
save("maxVolume.mat",'maxPop','maxPush');
writetable(table(maxPop',maxPush'),"maxVolume.xlsx");

function ind=During(init,Date,n)
    val=Date(init);
    for i=init:n
        if (Date(i)~=val)
            break
        end
    end
```

```
    if (i==n)
        ind=n;
    else
        ind=i-1;
    end
end
```

附录四：遗传退火 生成子图、计算、打印（结构代码相同省略）

CateEdge5

```
clc;clear;close all;
%对 DC5 的相邻节点做分类
%delete(gcp('nocreate'));
load Series.mat
load cate.mat
%p=parpool(12);

Normal_edge=[Count{1},Count{3},Count{6},Count{7}];
N=cell(1,4);
E=cell(1,4);
%% Node1DC5 左节点
%N{1}=Time_start(E{1})';
E{1}=find(Time_end==5)';
E{1}=intersect(E{1},Normal_edge);
N{1}=Time_start(E{1})';
%% Node2DC5 右节点
%N{2}=Time_end(E{2})';
E{2}=find(Time_start==5)';
E{2}=intersect(E{2},Normal_edge);
N{2}=Time_start(E{2})';
%% Node3DC5 左节点的右节点
N3=[];
for node=1:length(N{1})
    edge=find(Time_start==N{1}(node))';
    edge=intersect(edge,Normal_edge);
    N3=[N3,Time_end(edge)'];
    E{3}=[E{3},edge];
end
N3=N3(N3~=5);
N{3}=unique(N3);
E{3}=find(Time_end(E{3})~=5)';
clear node edge N3

%% Node4DC5 右节点的左节点
N4=[];
```

```
for node=1:length(N{2})
    edge=find(Time_end==N{2}(node))';
    edge=intersect(edge,Normal_edge);
    N4=[N4,Time_end(edge)'];
    E{4}=[E{4},edge];
end
N4=N4(N4~=5);
N{4}=unique(N4);
E{4}=find(Time_start(E{4})~=5)';
clear node edge N4
%% clear
%delete(p);
save('cateNode5.mat','N','E');
clear ans
```

CatePredict5.m

```
clc;clear;close all;
load FlowsAmong5.mat
%预测 DC5 附近的流量
delete(gcp('nocreate'));
p=parpool(8);
nearFlows=Flows(:,365*2-61+1:365*2);

clear i
len_pre=31;%预测长度
Predict=zeros(4,len_pre);
str={'左节点一月预测','右节点一月预测','左节点的右节点一月预测','右节点的左节点一月预测'};
err=zeros(1,4);
figure(2);
for i=1:4
    subplot(4,1,i);
    Data=nearFlows(i,:);
    len=length(Data);
    tmp_data=Data;
    for j=1:len_pre
        len_block=40;%滑块长度
        len=len+1;
        model = arima(5,0,1);
        fit= estimate(model,tmp_data(len-len_block:len-1)');
        %fit = estimate(model,tmp_data');
        fore = forecast(fit,1);
        tmp_data(len)=fore;
        %[a,b]=Smoothing(tmp_data,0.3);
        %tmp_data(len)=a+b;
```

```
        end
        hold on
        %bias=randn(1,31)*std(result{i})*0.5+exp(1:0.1:4);bias(1)=bias(1)+5;
        Predict(i,:)=tmp_data(len-len_pre+1:len);
        %plot([datetime(2022,11,12:30),datetime(2022,12,1:31)],Data(end-50+1:end),'b-');
        %plot(41:81,[tmp_data(end-len_proving+1:end),Predict(i,:)],'r+-');
        plot(datetime(2023,1,1:31),Predict(i,:),'r+-');
        xlim([datetime("2023-01-01") datetime("2023-01-31")]);
        ylabel("货量");
        title(str{i});
        %datetick('x','mm-dd');
        hold off
        clear len a b
end
PredictCate=Predict;
clear Predict
save("PredictCate5.mat","PredictCate");
delete(p);
clear ans

function y_pre=Average(y,n)
        m=length(y);
        y_pre=sum( y(m-n+1:m) ) / n;
end
function str=Node(node)
        str=strcat("DC",num2str(node));
end
```

FlowAmong5

```
clc;clear;close all;
%计算 DC5 附近的流量
%delete(gcp('nocreate'));
%p=parpool(12);
load Series.mat
load cateNode5.mat
startTime=min(find(Date==datetime(2022,11,1)));
During=days(datetime(2022,12,31)-datetime(2022,11,1))+1;
%Flows=zeros(4,During);
Flows=zeros(4,365*2);
hold on
for i=1:4
        %Flows(i,:)=sum(Time_var(E{i},end-During+1:end),1);
        Flows(i,:)=sum(Time_var(E{i},:),1);
        %plot([datetime(2022,11,1:30),datetime(2022,12,1:31)],Flows(i,:),'LineWidth',1);
```

```
    plot(1:365*2,Flows(i,:),'LineWidth',1);
end
clear i
grid on
legend("DC5 左节点","DC5 右节点","左节点的右节点","右节点的左节点");
%xlim([datetime("2022-11-1") datetime("2022-12-31")]);
ylabel("货量");
set(gca,'yscale','log');
title("DC5 最近二月的货物流量");
save("FlowsAmong5.mat","Flows");
hold off

%% clear
%delete(p);
clear ans
```

FlowsPredict5.m

```
clc;clear;close all;
load FlowsAmong5.mat
load cateNode5.mat
load Series.mat
%预测 DC5 附近的流量
delete(gcp('nocreate'));
p=parpool(12);
nearFlows=Flows(:,365*2-61+1:365*2);

clear i
len_pre=31;%预测长度
err=zeros(1,4);
Predict=cell(1,4);
for cate=1:4
    for i=1:length(E{cate})
        edge=E{cate}(i);
        Data=Time_var(edge,365*2-61+1:365*2);
        Data=filloutliers(Data,"linear");
        len=length(Data); clc;clear;close all;
```

GAforOrderIN5.m

```
load StatusAmong5.mat
load maxVolume.mat
delete(gcp('nocreate'));
%最后一位是值的大小
P=parpool(12);
Population=40;
```

```
Generation=200;
edge=Status{3}(:,1);
EncodingLength=length(edge)+1;
Genes=cell(1,31);
parfor day=1:31
    %solutionChange
    genes=cell(Generation,1);
    %Goal=zeros(1,Generation);
    tic
    t=0;
    genes{1}=GenerateParental2023(Population,EncodingLength,day);
    %top_gene=genes{1}{1};
    %Goal(1)=top_gene(1,EncodingLength);
    t=t+toc;
    disp(t);
    for k=2:Generation
        tic
        t=0;
        Parent=genes{k-1};%20
        Child1=Mate2023(Parent,3);%10
        Child2=Variate2023(Parent,0.1);%20
        genes{k}=Choose2023(Parent,Child1,Child2,day);
        t=t+toc;
        %Goal(k)=top_gene(1,EncodingLength);
        disp(t);
    end
    Genes{day}=genes;
end
%str=strcat(''Genes'',num2str(int(rand*100)),''.mat'');
save(''GenesIN5.mat'',''Genes'');
%delete(P);

function Genes=GenerateParental2023(Population,EncodingLength,day)
    %遗传算法生成亲代种群
    %Population 为种群大小,EncodingLength 为染色体基因长度,Genes 为种群染色体样本
    Genes=zeros(Population,EncodingLength);
    for p=1:Population
        Sequence=(rand(1,EncodingLength-1)<=0.7);   %生成个体基因序列
        [~,value,overflow]=SAforOrderIN5(Sequence,day);
        while (overflow==1)%不能存活的个体
            Sequence=(rand(1,EncodingLength-1)<=0.7);        %生成个体基因序列
[~,value,overflow]=SAforOrderIN5(Sequence,day);
        end
        Genes(p,:)=[Sequence,value];        %产生个体
```

```
    end
end

function Child=Mate2023(Parent,IntersectionNum)
    %遗传算法交配亲代得到子代
    %Parent 为亲代种群基因型,IntersectionNum 为交换基因的数量
    Child=Parent;
    [Population,EncodingLength]=size(Parent);
    EncodingLength=EncodingLength-1;
    for i=1:2:Population
        %利用 Logistics 混沌序列生成交叉基因座序列
        Locus0=rand;     Locus(1)=4*Locus0*(1-Locus0);
        for loc=2:IntersectionNum
            Locus(loc)=4*Locus(loc-1)*(1-Locus(loc-1));
        end
        Locus=1+ceil(EncodingLength*Locus);
        Locus=unique(Locus);
        %第 i 个个体与第 i+1 个个体交配,对应基因座交叉互换
        temp=Child(i,Locus);
        Child(i,Locus)=Child(i+1,Locus);
        Child(i+1,Locus)=temp;
    end
end

function varied_genotype=Variate2023(initial_genotype,Rate)
    %遗传算法亲代变异得到新的种群基因型
    %initial_genotype 为变异前种群基因型,varied_genotype 为变异后种群基因型,Rate 为变异率
    [Population,EncodingLength]=size(initial_genotype);
    EncodingLength=EncodingLength-1;
    Chosen=find(rand(1,Population)<Rate, 1);        %标记被变异的个体
    if (isempty(Chosen))                                    %如果都没人变异就找一个人变异
        Chosen=floor(Population*rand)+1;
    end
    varied_genotype=initial_genotype;
    numVaried=length(Chosen);
    for p=1:numVaried
        ind=GenerateBreakpoints2023(EncodingLength,3);   %使用三交换构造新的基因序列
        varied_genotype(Chosen(p),:)=varied_genotype(Chosen(p),[1:ind(1)-
1,ind(2)+1:ind(3),ind(1):ind(2),ind(3)+1:EncodingLength+1]);
    end
end
function ind=GenerateBreakpoints2023(range,num)
    %生成用于交换的 1*n 的系数向量
    rng("shuffle");
```

```
        ind=ceil(range*rand(1,num));
        ind=unique(ind);
        while (length(ind)<num)
            ind=ceil(range*rand(1,num));
            ind=unique(ind);
        end
    end
    function superior=Choose2023(A,B,C,day)
        %遗传算法选择优秀基因型作为新的亲本
        %A,B,C 分别为待选择的基因型
        Merge=[A;B;C];
        [Total,EncodingLength]=size(Merge);
        %EncodingLength=EncodingLength-1;
        %[~,ind1]=sort(Merge,2);          %整合三个种群
        sequenceVal=Merge(:,EncodingLength);
        for j=1:Total
            Sequence=Merge(j,:);
            chosen=nnz(Sequence);
            sequenceVal(j)=sequenceVal(j)*(1/chosen);
        end
        [~,ind2]=sort(sequenceVal,'descend');          %选择
        superior=Merge(ind2(1:40),:);
    end


            tmp_data=Data;
            len_block=40;%滑块长度
            for j=1:len_pre
                len=len+1;
                try
                model = arima(5,0,1);
                fit = estimate(model,tmp_data(len-len_block:len-1)');
                %fit = estimate(model,tmp_data');
                fore = forecast(fit,1);
                tmp_data(len)=fore;
                %[a,b]=Smoothing(tmp_data,0.3);
                %tmp_data(len)=a+b;
                catch
                    tmp_data(len)=Average(tmp_data,len_block);
                end
            end
            Predict{cate}(i,:)=tmp_data(len-len_pre+1:len);
            clear len a b edge
        end
    end
```

```
PredictFlows=Predict;
clear Predict
save("PredictFlows5.mat","PredictFlows");
delete(p);
clear ans

function y_pre=Average(y,n)
        m=length(y);
        y_pre=sum( y(m-n+1:m) ) / n;
end
```

SAforOrderIN5.m

```
%clc;clear;close all;
function [order,result,overFlow]=SAforOrderIN5(index,day)
        load StatusAmong5.mat
        load maxVolume.mat
        reOrder=0;
        rng('shuffle');
        num_node=81;
        %%init loading--E3
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        local=find(index==1);
        edge=Status{3}(local,1);
        start=Status{3}(local,2);
        term=Status{3}(local,3);
        contain=Status{3}(local,4);
        predict=Status{3}(local,5+day-1);
        predict=min(predict,contain);
        predict=Clean(predict);
        %rate=sum(predict);%正在跑的总流量
        Total=sum(Status{1}(:,4));%流向 DC5 的需要分配的流量
        num_edge=length(edge);%E3 集合的大小
        shutPop=ones(1,num_node);%一开始所有出点都开着
        shutPush=ones(1,num_node);%一开始所有入点都开着
        overFlow=0;%流量是否溢出
        %%产生一组可接受的初始解
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %order=predict*CalcPort(total,rate);
        reOrder=0;
        order=zeros(1,num_edge);
        order(:)=Total/num_edge;
        for i=1:num_edge%考虑 E3 的运输能力
                if (predict(i)+order(i)>contain(i))
                        reOrder=predict(i)+order(i)-contain(i);%把边能多承载的流量拿出来
```

```
            order(i)=contain(i)-predict(i);%把边塞满
            if (order(i)<0)
                error('错误的分配');
            end
            if (predict(i)+order(i)>contain(i))
                error('错误的分配');
            end
        else%这条边没有塞满
            if (reOrder<=(contain(i)-(predict(i)+order(i))+1)/2)
                order(i)=order(i)+reOrder;
                reOrder=0;
            else
                order(i)=order(i)+(contain(i)-(predict(i)+order(i))+1)/2;
                reOrder=reOrder-(contain(i)-(predict(i)+order(i))+1)/2;
            end
        end
    end
    flag=1;%还存在有容量的边;
    while (reOrder>0 && flag==1)
        flag=0;
        for i=1:num_edge%考虑 E3 的运输能力
            if (predict(i)+order(i)<contain(i))
                flag=1;
                if (reOrder<=(contain(i)-(predict(i)+order(i))+1)/2)
                    order(i)=order(i)+reOrder;
                    reOrder=0;
                else
                    if (reOrder>0)
                    order(i)=order(i)+(contain(i)-(predict(i)+order(i))+1)/2;
                    reOrder=reOrder-(contain(i)-(predict(i)+order(i))+1)/2;
                    end
                end
            end
        end
    end
    if (flag==0 && reOrder>0)
        overFlow=1;
        result=-1;
        return
    end
    clear flag

    %考虑 N1 的出货能力和 N3 的收获能力
    runningPop=zeros(1,num_node);
```

```
runningPush=zeros(1,num_node);
for i=1:num_edge
    node=start(i);%取出 N1 中的点
    runningPop(node)=runningPop(node)+predict(i)+order(i);
    node=term(i);%取出 N3 中的点
    runningPush(node)=runningPush(node)+predict(i)+order(i);
    clear node
end
for i=1:num_edge%处理点的冲突
    if (shutPop(start(i))==0 || shutPush(term(i))==0)
        continue
    end
    node=start(i);%N1
    if (runningPop(node)>maxPop(node))
        if (runningPop(node)-maxPop(node)<=order(i))
            take=runningPop(node)-maxPop(node);
            reOrder=reOrder+take;
            order(i)=order(i)-take;
            runningPop(node)=runningPop(node)-take;
            shutPop(node)=0;%冲突已经解决关闭冲突节点
        else
            take=order(i);
            reOrder=reOrder+take;
            order(i)=order(i)-take;
            runningPop(node)=runningPop(node)-take;
        end
    end
    clear node take
    node=term(i);%N1
    if (runningPush(node)>maxPush(node))
        if (runningPush(node)-maxPush(node)<=order(i))
            take=runningPush(node)-maxPush(node);
            reOrder=reOrder+take;
            order(i)=order(i)-take;
            runningPush(node)=runningPush(node)-take;
            shutPush(node)=0;%冲突已经解决关闭冲突节点
        else
            take=order(i);
            reOrder=reOrder+take;
            order(i)=order(i)-take;
            runningPush(node)=runningPush(node)-take;
        end
    end
    clear node take
```

```
end

%循环解决冲突
while (reOrder>0 && (nnz(shutPop)+nnz(shutPush))>0)%还有未分配的流量并且不能所有节点都关闭
了
    for i=1:num_edge%考虑 E3 的运输能力
        if (shutPop(start(i))==0 || shutPush(term(i))==0)
            continue
        end
        if (predict(i)+order(i)>contain(i))
            reOrder=predict(i)+order(i)-contain(i);%把边能多承载的流量拿出来
            order(i)=contain(i)-predict(i);%把边塞满
        else%这条边没有塞满
            if (reOrder<=(contain(i)-(predict(i)+order(i))+1)/2)
                order(i)=order(i)+reOrder;
                reOrder=0;
            else
                order(i)=order(i)+(contain(i)-(predict(i)+order(i))+1)/2;
                reOrder=reOrder-(contain(i)-(predict(i)+order(i))+1)/2;
            end
        end
    end
    flag=1;%还存在有容量的边;
    while (reOrder>0 && flag==1)
        flag=0;
        for i=1:num_edge%考虑 E3 的运输能力
            if (shutPop(start(i))==0 || shutPush(term(i))==0)
                continue
            end
            if (predict(i)+order(i)<contain(i))
                flag=1;
                if (reOrder<=(contain(i)-predict(i)+1)/2)
                    order(i)=order(i)+reOrder;
                    reOrder=0;
                else
                    if (reOrder>0)
                    order(i)=order(i)+(contain(i)-predict(i)+1)/2;
                    reOrder=reOrder-(contain(i)-predict(i)+1)/2;
                    end
                end
            end
        end
    end
```

```
    if (flag==0 && reOrder>0)
        overFlow=1;
        result=-1;
        return
    end
    clear flag

%考虑 N1 的出货能力和 N3 的收获能力
runningPop=zeros(1,num_node);
runningPush=zeros(1,num_node);
for i=1:num_edge
    if (shutPop(start(i))==0 || shutPush(term(i))==0)
        continue
    end
    node=start(i);%取出 N1 中的点
    runningPop(node)=runningPop(node)+predict(i)+order(i);
    node=term(i);%取出 N3 中的点
    runningPush(node)=runningPush(node)+predict(i)+order(i);
    clear node
end
    for i=1:num_edge%处理点的冲突
        if (shutPop(start(i))==0 || shutPush(term(i))==0)
            continue
        end
        node=start(i);%N1
        if (runningPop(node)>maxPop(node))
            if (runningPop(node)-maxPop(node)<=order(i))
                take=runningPop(node)-maxPop(node);
                reOrder=reOrder+take;
                order(i)=order(i)-take;
                runningPop(node)=runningPop(node)-take;
                shutPop(node)=0;%冲突已经解决关闭冲突节点
            else
                take=order(i);
                reOrder=reOrder+take;
                order(i)=order(i)-take;
                runningPop(node)=runningPop(node)-take;
            end
        end
        clear node take
        node=term(i);%N1
        if (runningPush(node)>maxPush(node))
            if (runningPush(node)-maxPush(node)<=order(i))
                take=runningPush(node)-maxPush(node);
```

```
                reOrder=reOrder+take;
                order(i)=order(i)-take;
                runningPush(node)=runningPush(node)-take;
                shutPush(node)=0;%冲突已经解决关闭冲突节点
            else
                take=order(i);
                reOrder=reOrder+take;
                order(i)=order(i)-take;
                runningPush(node)=runningPush(node)-take;
            end
        end
        clear node take
    end
end

%%退火,order 为分配方案
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
order=Clean(order);
begin_heat=100;
end_heat=1;
Heat=begin_heat;
Rate=0.9;
MarkovLength=500;
totalContain=Status{3}(:,4);%全部边为了得到目标量
totalPredict=Status{3}(:,5+day-1);
totalLen=length(totalPredict);
changed=Effected(totalPredict,order,local);
Res=CalcGoal(changed,totalContain,totalLen);
for Layer=1:MarkovLength
    upFlow=Generate(predict,contain,order);
    tentative=1;%转移试探次数
    while (upFlow(1)==-1 || CheckValid(order+upFlow,start,term,maxPop,maxPush,contain)==0)
        upFlow=Generate(predict,contain,order);
        tentative=tentative+1;
        if (tentative>1000)
            result=Res;
            return
        end
    end
    clear tentative
    changed=Effected(totalPredict,order+upFlow,local);
    tmp_Res=CalcGoal(changed,totalContain,totalLen);
    cost=tmp_Res-Res;
    if (cost>0 || exp(-cost/Heat)>rand)
```

```
                order=order+upFlow;
                Res=tmp_Res;
            end
            Heat=Heat*Rate;
            if (Heat<end_heat)
                break
            end
        end
        result=Res;
end

function rate=CalcPort(total,running)
        rate=total/running;
end
function valid=CheckValid(runningFlow,start,term,maxPop,maxPush,edgeContain)
        len=length(runningFlow);
        num=81;
        valid=1;
        runningPop=zeros(1,num);
        runningPush=zeros(1,num);
        for i=1:len
            if (runningFlow(i)>edgeContain(i))
                valid=0;
                return
            end
            s=start(i);
            t=term(i);
            runningPop(s)=runningPop(s)+runningFlow(i);
            runningPush(t)=runningPush(t)+runningFlow(i);
        end
        for i=1:num
            if (runningPop(i)>maxPop(i))
                valid=0;
                return;
            end
            if (runningPush(i)>maxPush(i))
                valid=0;
                return;
            end
        end
end
function upFlow=Generate(predict,contain,res)
        res=res';
        running=predict+res;%在当前分配方案下的流量
```

```
running=Clean(running);
n=length(running);
release=contain-running;
release=Clean(release);
if (~isempty(find(release < 0)))
    upFlow=-ones(1,n);
    return;
    %error("分配了大于线路容量的流量");
end
[sorted_release,ind_sort]=sort(release);
shake=0;
for i=1:n
    if (shake>=0)
        upFlow(ind_sort(i))=sorted_release(i)*0.1;
        shake=shake+upFlow(ind_sort(i));
    else
        upFlow(ind_sort(i))=-min(sorted_release(i)*0.1,res(i));
        shake=shake+upFlow(ind_sort(i));
    end
end
clear i
p=1;
while(shake>0)
    if (shake<1 && res(ind_sort(p))+upFlow(ind_sort(p))>shake)%消除抖动循环
        upFlow(ind_sort(p))=upFlow(ind_sort(p))-shake;
        shake=0;
        break
    end
    take=shake*0.2;
    while (res(ind_sort(p))+upFlow(ind_sort(p))-take<=0)%不能够被调节
        take=take*0.8;
        if (take<1)
            overFlow=1;
            return
        end
    end
    upFlow(ind_sort(p))=upFlow(ind_sort(p))-take;
    shake=shake-take;
    p=mod(p,n)+1;
end
while(shake<0)
    if (shake>-1 && res(ind_sort(p))+upFlow(ind_sort(p))-shake<=contain(ind_sort(p)))%消除抖动循
环
        upFlow(ind_sort(p))=upFlow(ind_sort(p))-shake;%减负等于加正下同理;
```

```
                shake=0;
                break
            end
            take=shake*0.2;
            if (res(ind_sort(p))+upFlow(ind_sort(p))+take<=contain(ind_sort(p)))%能够被调节
                upFlow(ind_sort(p))=upFlow(ind_sort(p))-take;
                shake=shake-take;
            end
            p=mod(p,n)+1;
        end
    end
    function Res=Effected(raw,var,index)
        len=length(var);
        for i=1:len
            raw(index(i))=raw(index(i))+var(i);
        end
        Res=raw;
    end
    function loading=CalcLoad(contain,running)
        loading=running./contain;
    end
    function Res=CalcGoal(running,contain,totalLen)
        loading=CalcLoad(contain,running);
        Average=sum(loading)/totalLen;
        Res=sum((loading-Average).^2)/totalLen;
        Res=1/Res;
    end
    function Res=Clean(raw)
        Res=fix(raw);
    End
```

Weight.m

```
%clc;clear;close all;
function [weightNode,weightEdge]=Weight(start,term,value)
weightEdge=zeros(1049,8);%8 个指标
weightNode=zeros(81,12);%12 个指标
for edge=1:num_connect
    weightEdge(edge,:)=MeasureEdge(edge,start,term,value);
end
for i=1:8
    weightEdge(:,i)=zscore(weightEdge(:,i));
end
weightEdge(:,8)=-weightEdge(:,8);%历史最长流量不存在天数是极小指标
indEdge_clean=find(weightEdge(1,:));
```

```
tmp=weightEdge(indEdge_clean,:);
weightEdge=[];
weightEdge=tmp;
clear i edge tmp
for node=1:num
```

MeasureNode.m

```
weightNode(node,:)=MeasureNode(node,start,term,value);
end
for i=1:12
    weightNode(:,i)=zscore(weightNode(:,i));
end
indNode_clean=find(weightNode(1,:));
tmp=weightNode(indNode_clean,:);
weightNode=[];
weightNode=tmp;
clear i node tmp
end
```

Score.m

```
function [scoreNode,scoreEdge]=Score(weightNode,weightEdge)
    evalNode(1,:)=[0.154,0.155,0.123,0.125,0.133,0.09,0.1,0.166,0.163];
    evalNode(2,:)=[-0.197,-0.201,-0.021,-0.042,-0.065,0.571,0.537,-0.046,-0.122];
    evalNode(3,:)=[-0.182,-0.153,0.571,0.561,-0.337,0.008,-0.136,-0.098,-0.091];
    evalPartN=[-0.65887,0.1548,0.1311];
    for i=1:81
        nodeY=zeros(1,3);
        for j=1:3
            nodeY(j)=evalNode(j).*weightNode(i);
        end
        scoreNode(i)=-sum(nodeY.*evalPartN);
    end
    evalEdge(1,:)=[0.225,0.219,0.257,0.217,0.261,0.138,0.019];
    evalEdge(2,:)=[0.409,0.432,-0.216,0.319,-0.280,0.13,0.381];
    evalEdge(3,:)=[0.064,0.077,-0.119,0.244,-0.161,0.584,0.745];
    evalPartE=[0.4775,0.1923,0.14185];
    for i=1:1049
        edgeY=zeros(1,3);
        for j=1:3
            edgeY(j)=evalEdge(j).*weightEdge(i);
        end
        scoreEdge(i)=-sum(edgeY.*evalPartE);
    end
```

```
end

function Value=MeasureEdge(edge,Time_start,Time_end,Time_var)
    %Col1:最大流量  Col2:最小流量   Col3:平均流量
    %Col4:最长流量存在天数  Col5:最长流量不存在天数   Col6:流量存在天数
    %Col7:该边流量占左节点的总发货路径数比  Col8:该边流量占右节点的总收货流量比
    leftNode=Time_start(edge);
    leftEdge=find(Time_start==leftNode);
    rightNode=Time_end(edge);
    rightEdge=find(Time_end==rightNode);
    Value=zeros(1,8);
    Value(3)=mean(Time_var(edge,:));
    Value(6)=nnz(Time_var(edge,:));
    Value(7)=Value(7)+sum(sum(Time_var(leftEdge,:),1));
    Value(8)=Value(8)+sum(sum(Time_var(rightEdge,:),1));
    running=0;%当天是否有流量在跑,计算存在天数
    dayFlag=1;
    for day=1:365*2
        Value(1)=max(Value(1),Time_var(edge,day));
        Value(2)=min(Value(2),Time_var(edge,day));
        if (Time_var(edge,day)>0)
            if (running==0)
                Value(5)=max(Value(5),day-dayFlag);
                running=1;
                dayFlag=day;
            end
        else
            if (running==1)
                Value(4)=max(Value(4),day-dayFlag);
                running=0;
                dayFlag=day;
            end
        end
    end
    if (running==0)%统计最后没有更新的序列状态
        Value(5)=max(Value(5),day-dayFlag);
    else
        Value(4)=max(Value(4),day-dayFlag);
    end
    if (Value(7)~=0)
        Value(7)=Value(3)/Value(7);%流量占比
    end
    if (Value(8)~=0)
        Value(8)=Value(3)/Value(8);%流量占比
```

```
    end
end
```

MeasureNode.m

```
function Value=MeasureNode(node,Time_start,Time_end,Time_var)
    %Col1:最大收货量  Col2:最小收货量   Col3:平均收货量
    %Col4:最大发货量  Col5:最小发货量   Col6:平均发货量
    %Col7:收货总天数  Col8:发货总天数   Col9:同时发货总天数
    %Col9:最大邻接点数      Col10:最小邻接点数  Col12:平均邻接点数
    Value=zeros(1,12);
    leftEdge=find(Time_end==node);%寻找与左节点相邻个边
    rightEdge=find(Time_start==node);%寻找与右节点相邻个边
    for day=1:365*2
        tmpValue=zeros(1,3);
        %Col1 收货量
        %Col2 发货量
        %Col3 邻接点数
        for edge=1:length(leftEdge)%计算收货指标:1+3
            tmpValue(1)=tmpValue(1)+Time_var(edge,day);
            tmpValue(3)=tmpValue(3)+(Time_var(edge,day)>=0);
        end
        for edge=1:length(rightEdge)%计算发货指标:2+3
            tmpValue(2)=tmpValue(2)+Time_var(edge,day);
            tmpValue(3)=tmpValue(3)+(Time_var(edge,day)>=0);
        end
        Value(1)=max(Value(1),tmpValue(1));
        Value(2)=min(Value(2),tmpValue(1));
        Value(3)=Value(3)+tmpValue(1);
        Value(4)=max(Value(4),tmpValue(2));
        Value(5)=min(Value(5),tmpValue(2));
        Value(6)=Value(6)+tmpValue(2);
        Value(7)=Value(7)+(tmpValue(1)>0);
        Value(8)=Value(8)+(tmpValue(2)>0);
        Value(9)=Value(9)+((tmpValue(1)+tmpValue(2))>0);
        Value(10)=max(Value(10),tmpValue(3));
        Value(11)=min(Value(11),tmpValue(3));
        Value(12)=Value(12)+tmpValue(3);
    end
    if (Value(7)~=0)
        Value(3)=Value(3)/Value(7);
    end
    if (Value(8)~=0)
        Value(6)=Value(6)/Value(8);
```

```
    end
    if (Value(9)~=0)
        Value(12)=Value(12)/Value(9);
    end
end
```

GAforAdd.m

```
clc;clear;close all;
load Series.mat
load predictResult.mat
%delete(gcp('nocreate'));
%最后一位是值的大小
%P=parpool(12);
Population=400;
Generation=200;
EncodingLength=81*2;
%disp(toOrder_all);
%solutionChange
Genes=cell(1,4);
global c2;
global c3;
global c1;
global yita;
c2=0.3+rand*0.4;
c3=0.3+rand*0.4;
c1=(c2+c3)*1000;
yita=4/9;
%%
idx=0;
for i=1:7:21
    idx=idx+1;
    Flow(idx,:)=sum(Predict(i:i+7,:));
    Flow(idx,:)= filloutliers(Flow(idx,:),"linear");
end
clear idx;

%% GAforAdd
for week=1:4
    running=Flow(week);
    genes=cell(Generation,1);
    tic
    t=0;
    genes{1}=GenerateParental2023(Population,EncodingLength);
```

```
%top_gene=genes{1}{1};
%Goal(1)=top_gene(1,EncodingLength);
      t=t+toc;
      disp(t);
      for k=2:Generation
            tic
           t=0;
    Parent=genes{k-1};%20
    Child1=Mate2023(Parent,3);%10
    Child2=Variate2023(Parent,0.1);%20
    genes{k}=Choose2023(Parent,Child1,Child2,Population,running);
    t=t+toc;
    Genes{week}=genes;
    disp(t);
end
end
%str=strcat(''Genes'',num2str(int(rand*100)),''.mat'');
save(''GenesAdd.mat'',''Genes'');
%delete(P);

function Genes=GenerateParental2023(Population,EncodingLength,day,toOrder)
    %遗传算法生成亲代种群
    %Population 为种群大小,EncodingLength 为染色体基因长度,Genes 为种群染色体样本
    Genes=zeros(Population,EncodingLength);
    for p=1:Population
        Sequence=(rand(1,EncodingLength)<=0.09);  %生成个体基因序列
        while (nnz(Sequence)>15)%约束条件,不能存活的个体
            Sequence=(rand(1,EncodingLength)<=0.09);  %生成个体基因序列
        end
        Genes(p,:)=Sequence;       %产生个体
    end
end

function Child=Mate2023(Parent,IntersectionNum)
    %遗传算法交配亲代得到子代
    %Parent 为亲代种群基因型,IntersectionNum 为交换基因的数量
    Child=Parent;
    [Population,EncodingLength]=size(Parent);
    EncodingLength=EncodingLength-1;
    for i=1:2:Population
        %利用 Logistics 混沌序列生成交叉基因座序列
        Locus0=rand;    Locus(1)=4*Locus0*(1-Locus0);
        for loc=2:IntersectionNum
            Locus(loc)=4*Locus(loc-1)*(1-Locus(loc-1));
```

```
            end
            Locus=1+ceil(EncodingLength*Locus);
            Locus=unique(Locus);
            %第 i 个个体与第 i+1 个个体交配,对应基因座交叉互换
            temp=Child(i,Locus);
            Child(i,Locus)=Child(i+1,Locus);
            Child(i+1,Locus)=temp;
            while (nnz(Child(i,:))>15)%约束条件,不能存活的个体
                        tmp=Child(i,:);
                        ind=find(tmp==1);
                        ind=ind(ceil(rand*length(ind)));
                        tmp(ind)=0;
                        Child(i,:)=tmp;
            end
            while (nnz(Child(i+1,:))>15)%约束条件,不能存活的个体
                        tmp=Child(i+1,:);
                        ind=find(tmp==1);
                        ind=ind(ceil(rand*length(ind)));
                        tmp(ind)=0;
                        Child(i+1,:)=tmp;
            end
      end
end

function varied_genotype=Variate2023(initial_genotype,Rate)
      %遗传算法亲代变异得到新的种群基因型
      %initial_genotype 为变异前种群基因型,varied_genotype 为变异后种群基因型,Rate 为变异率
      [Population,EncodingLength]=size(initial_genotype);
      EncodingLength=EncodingLength-1;
      Chosen=find(rand(1,Population)<Rate, 1);        %标记被变异的个体
      if (isempty(Chosen))                                                %如果都没人变异就找一个人变异
            Chosen=floor(Population*rand)+1;
      end
      varied_genotype=initial_genotype;
      numVaried=length(Chosen);
      for p=1:numVaried
                  ind=GenerateBreakpoints2023(EncodingLength,3);   %使用三交换构造新的基因序列
                  varied_genotype(Chosen(p),:)=varied_genotype(Chosen(p),[1:ind(1)-
1,ind(2)+1:ind(3),ind(1):ind(2),ind(3)+1:EncodingLength+1]);
                  while (nnz(varied_genotype(Chosen(p),:))>15)%约束条件,不能存活的个体
                        tmp=varied_genotype(Chosen(p),:);
                        ind=find(tmp==1);
                        ind=ind(ceil(rand*length(ind)));
                        tmp(ind)=0;
```

```
                varied_genotype(Chosen(p),:)=tmp;
            end
        end
    end
    function ind=GenerateBreakpoints2023(range,num)
        %生成用于交换的 1*n 的系数向量
        rng("shuffle");
        ind=ceil(range*rand(1,num));
        ind=unique(ind);
        while (length(ind)<num)
            ind=ceil(range*rand(1,num));
            ind=unique(ind);
        end
    end
    function superior=Choose2023(A,B,C,Population,running)
        %遗传算法选择优秀基因型作为新的亲本
        %A,B,C 分别为待选择的基因型
        Merge=[A;B;C];
        [Total,EncodingLength]=size(Merge);
        tmp_order=zeros(1,EncodingLength);
        for j=1:Total
            Sequence=Merge(j,:);
            load(j)=Calccost(Sequence,running);
        end
        [~,ind2]=sort(load,'descend');              %选择
        superior=Merge(ind2(1:Population),:);
    end

    function Res=Calccost(Sequence,running)
        global c1;
        global c2;
        global c3;
        global yita;
        num=nnz(Sequence);
        %c1 开设成本,c2 运输成本,c3 处理成本
        reOrder=sum(running)*0.02;
        reOrder=reOrder*(1+rand*0.4-0.2);
        Load=mean(CalcLoad(Sequence,reOrder,running));
        Res=Load/(c1*num+(c2+c3)*reOrder/yita);
    end
```