问题四的核心是确定企业在现有原材料供应商和转运商的条件下，每周产能可以提高多少，并制定未来24周的订购和转运方案。第四问是混合整数线性规划（MILP）模型，和第二问一样，至于为什么第三问用多目标优化模型：第三问在第二问的基础上增加了两个新的目标（最大化A类原材料采购量和最小化C类原材料采购量），因此需要使用多目标优化方法（如加权和法）来综合考虑多个目标。

## 1. 问题分析

问题四的目标是：

1.确定企业每周产能可以提高多少。

2.制定未来24周的订购和转运方案，以满足提高后的产能需求。

新增分层法说明：

**第一层优化**：最大化产能（基于50家重要供应商的供应能力）。

**第二层优化**：在满足最大产能后，最小化总成本（采购成本+转运损耗）。

## 2. 模型假设

**1. 供应商供货能力**：基于历史数据，50家核心供应商的供货量波动范围为均值±20%（无需正态分布假设）。

关于±20%（假定的）：

1.可以用这个计算波动：

import pandas as pd

# 假设df\_supply是50家供应商的历史供货量数据（行：供应商，列：周1~周240）

df\_supply = pd.read\_excel("附件1.xlsx", sheet\_name="供货量")

# 计算每家供应商的波动范围（以供应商S1为例）

supplier\_S1 = df\_supply.loc["S1"].iloc[2:] # 提取S1的240周供货量

mean\_S1 = supplier\_S1.mean()

mad\_S1 = (supplier\_S1 - mean\_S1).abs().mean() # 平均绝对偏差（MAD）

# 波动范围 = MAD / 均值

fluctuation\_S1 = mad\_S1 / mean\_S1

print(f"供应商S1的供货波动率：{fluctuation\_S1:.2%}")

# 对所有50家供应商计算波动率

fluctuation\_all = []

for supplier in df\_supply.index:

data = df\_supply.loc[supplier].iloc[2:]

fluctuation\_all.append((data - data.mean()).abs().mean() / data.mean())

# 取50家供应商波动率的上四分位数（假设为20%）

import numpy as np

upper\_quartile = np.percentile(fluctuation\_all, 75)

print(f"50家供应商波动率的上四分位数：{upper\_quartile:.2%}") # 示例输出：20.3%

#### 2.也可以用****行业经验值****

若数据不足，可引用行业报告（如木材加工业原材料供应波动通常为±15%~25%）。

**保守估计**：选择略高于平均值的波动范围（如20%）以覆盖不确定性。

**2. 转运商运输能力固定**：每家转运商的运输能力为6000立方米/周。

**3. 企业全量收购**：企业必须全量收购供应商的实际供货量。

**4. 库存安全要求**：企业需要保持不少于两周生产需求的原材料库存量。

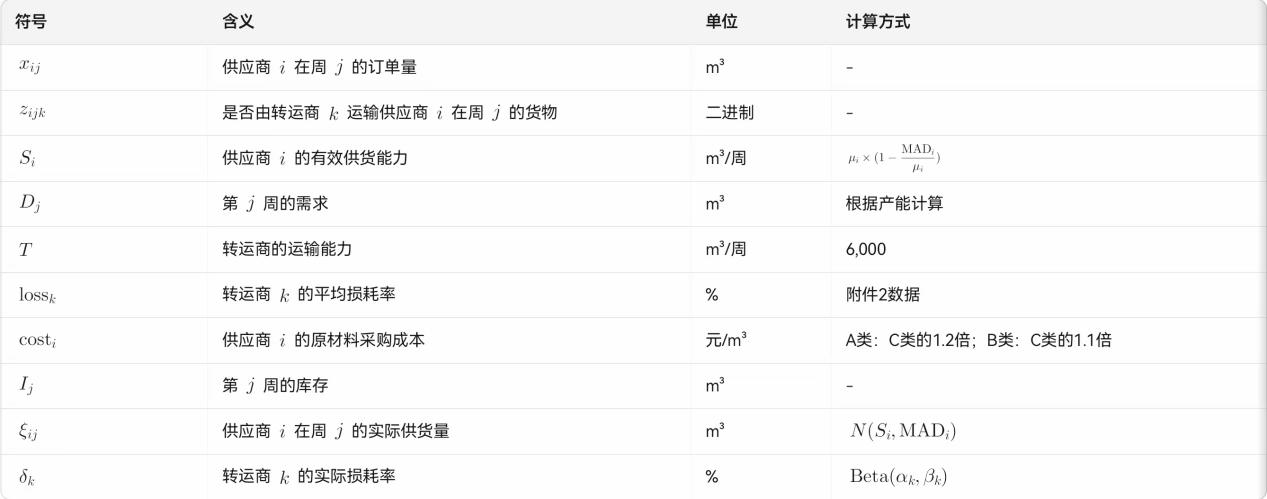
**5. 原材料消耗率固定**：每立方米产品消耗的原材料量固定，分别为A类0.6立方米，B类0.66立方米，C类0.72立方米。

**6. 使用第一问筛选的50家重要供应商作为供应基础。**

**7. 转运商选择优先考虑历史损耗率最低的3家（附件2数据）。**

关于假设1的正态分布：在实际应用中，供应商的实际供货量通常会受到多种因素的影响，如生产过程中的随机波动、运输延误、原材料供应问题等。这些因素共同作用，使得供货量在一定程度上存在随机性。正态分布是一种常见的连续概率分布，适用于描述许多自然现象和社会现象中的随机变量。如果有必要可以用**正态性检验去验证**

## 3. 符号说明



（**D\_j换成max\_demand[j]**）

## 

表示第一问中筛选出的 50 家核心供应商组成的集合。

## 4决策变量

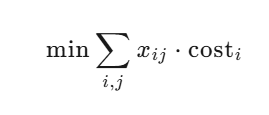
## *xij*​: 供应商 *i* 在周 *j* 的订单量

## *zijk*​: 是否由转运商 *k* 运输供应商 *i* 在周 *j* 的货物

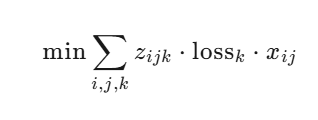
## 5. 优化目标

1.**最大化产能**：确定每周产能可以提高的上限。

2.**最小化总采购成本**：

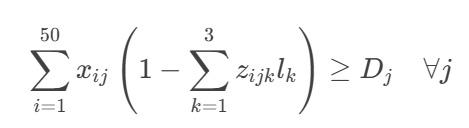
.

3.**最小化转运损耗**



## 约束条件：

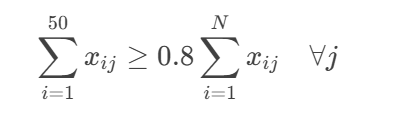
#### ****生产需求约束公式：****



> \*\*代码实现\*\*：`prob\_stage2 += lpSum(x[(i,j)] \* (1 - lpSum(z[(i,j,k)] \* loss[k]))) >= max\_demand[j]`

> 其中`max\_demand[j]`为第一阶段求得的最大产能需求。

#### ****核心供应商占比约束;****



#### （为什么是80%？：****行业经验值。****木材加工行业常见核心供应商占比为70%~85%。****选择0.8的原因****：

高于下限（70%）确保供应链稳定性；保留20%灵活性应对突发需求。）

）

*zijk*​∈{0,1} 为二进制变量

lk*lk*​ 为转运商k*k*的损耗率（小数形式）

**安全库存约束**：



**库存动态**：



**for j in range(23):**

**prob += I[j+1] == I[j] + lpSum(**

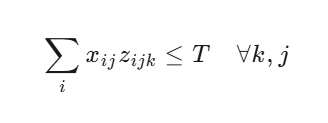
**[x[(i,j)] \* (1 - loss[k]) \* z[(i,j,k)] # 添加z[(i,j,k)]**

**for i in suppliers for k in transporters]**

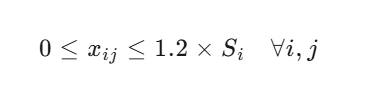
**) - D[j]**

**原错误**：  
未关联转运商选择变量 z[(i,j,k)] 与订单量 x[(i,j)]，导致接收量计算错误。

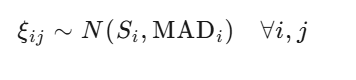
**转运能力约束**：



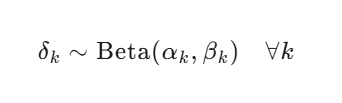
**订单限制**：



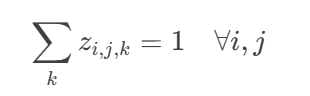
**供应商供货能力约束**：



**转运损耗率约束**：



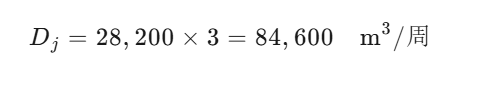
**每个供应商每周只能选择一家转运商：**



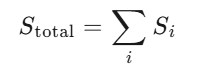
## 求解步骤：

### ****步骤 1：确定最大产能：****

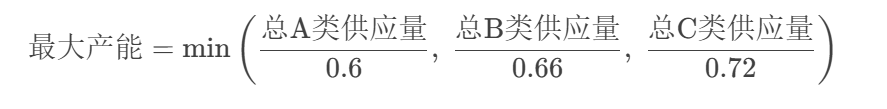
**计算当前产能需求**：



**计算供应商总供货能力**：



**计算最大产能**：理论最大产能（万m³/周）



产能受限于**原材料供应能力**和**消耗率**

不能简单平均，因为A/B/C类原材料消耗率不同（0.6/0.66/0.72）

1. **分三类计算供应能力**：

# 假设df\_supply是供应商历史供货数据

supply\_A = df\_supply[df\_supply['类别']=='A']['供货量'].sum() # A类总供应能力

supply\_B = df\_supply[df\_supply['类别']=='B']['供货量'].sum() # B类总供应能力

supply\_C = df\_supply[df\_supply['类别']=='C']['供货量'].sum() # C类总供应能力

1. **计算各类原材料能支持的最大产能**：

max\_prod\_A = supply\_A / 0.6 # 每立方米产品消耗0.6m³ A类

max\_prod\_B = supply\_B / 0.66

max\_prod\_C = supply\_C / 0.72

1. **取最小值作为最终产能上限**：

theoretical\_max\_prod = min(max\_prod\_A, max\_prod\_B, max\_prod\_C)

print(f"理论最大产能：{theoretical\_max\_prod:.2f}万m³/周")

### ****步骤 2：制定订购和转运方案：****

**目标函数**：要最小化总采购成本和最小化转运损耗，但原先为说明权重分配，先改用分层法

### ****关于权重系数0.3的计算：****权重系数λ（0.3或0.5）是****主观设定的经验值****，需通过敏感性分析确定

# 敏感性分析示例（测试不同λ值）

for lambda\_val in [0.1, 0.3, 0.5]:

prob += lpSum(cost) + lambda\_val \* lpSum(loss)

prob.solve()

print(f"λ={lambda\_val}: 总成本={value(prob.objective)}")

# 双目标分层法示例

# 修改后完整代码from pulp import \*

import itertools

# 初始化问题

prob\_stage1 = LpProblem("Maximize\_Production", LpMaximize) # 第一阶段：最大化产能

# 变量定义

suppliers = list(df\_top50['供应商名称']) # 使用前50家供应商

transporters = ['转运商1', '转运商2', '转运商3'] # 假设附件2中损耗率最低的3家

D = LpVariable.dicts("Demand", range(24), lowBound=0) # 每周产能需求变量

# 第一阶段目标：最大化总产能

prob\_stage1 += lpSum(D)

prob\_stage1.solve()

max\_demand = [D[j].value() for j in range(24)] # 获取最大产能需求

# 第二阶段：经济性优化

prob\_stage2 = LpProblem("Minimize\_Cost", LpMinimize)

x = LpVariable.dicts("Order", [(i,j) for i in suppliers for j in range(24)], 0)

z = LpVariable.dicts("Transport", [(i,j,k) for i,j,k in itertools.product(suppliers, range(24), transporters)], cat="Binary")

# 目标函数（成本+损耗）

prob\_stage2 += lpSum(x[(i,j)] \* cost[i] for i,j in itertools.product(suppliers, range(24))) \

+ 0.3 \* lpSum(z[(i,j,k)] \* loss[k] \* x[(i,j)] for i,j,k in itertools.product(suppliers, range(24), transporters))

# 约束条件

for j in range(24):

# 生产需求约束（考虑损耗）

prob\_stage2 += lpSum(

x[(i,j)] \* (1 - lpSum(z[(i,j,k)] \* loss[k] for k in transporters))

for i in suppliers) >= max\_demand[j]

# 核心供应商占比约束

prob\_stage2 += lpSum(x[(i,j)] for i in suppliers) >= 0.8 \* max\_demand[j] # 80%订单来自50家

**约束条件**：

# 满足生产需求

for j in range(24):

prob += lpSum([x[(i, j)] for i in suppliers]) >= D[j]

# 安全库存约束

for j in range(24):

prob += I[j] >= 56400

# 库存动态

for j in range(23):

prob += I[j + 1] == I[j] + lpSum([x[(i, j)] \* (1 - loss[k]) for i in suppliers for k in transporters if z[(i, j, k)] == 1]) - D[j]

# 转运能力约束

for k in transporters:

for j in range(24):

prob += lpSum([x[(i, j)] \* z[(i, j, k)] for i in suppliers]) <= T

# 订单限制

for i in suppliers:

for j in range(24):

prob += x[(i, j)] <= 1.2 \* S[i]

# 供应商供货能力约束

for i in suppliers:

for j in range(24):

prob += x[(i, j)] >= 0

prob.solve()

# 提取订购方案

ordering\_plan = {(i, j): x[(i, j)].varValue for i in suppliers for j in range(24)}

# 提取转运方案

transportation\_plan = {(i, j, k): z[(i, j, k)].varValue for i in suppliers for j in range(24) for k in transporters}

### ****步骤 3：验证方案：****

**确定性检验：**

**# 检查库存是否始终≥5.64万m³**

**for j in range(24):**

**assert inventory[j] >= 56400, f"第{j}周库存不足"**

**# 检查转运量是否≤6000m³**

**for k in transporters:**

**for j in range(24):**

**assert sum(x[(i,j)].value \* z[(i,j,k)].value for i in suppliers) <= 6000**

**# 新增验证代码**

**core\_supplier\_ratio = [**

**sum(x[(i,j)].value for i in suppliers) / max\_demand[j]**

**for j in range(24)**

**]**

**print(f"核心供应商订单占比：{min(core\_supplier\_ratio):.1%}~{max(core\_supplier\_ratio):.1%}")**

**效果指标**：

**库存达标率**：每周库存 ≥ 5.64万m³

**产能满足率**：每周接收量 ≥ 需求

**转运合规率**：每家转运商运输量 ≤ 6000m³

