生产过程中的决策问题

摘要

对于问题一,

关键字：

一、 问题重述

1.1 问题背景

为高效利用有效资源、减少生产成本、提高运营收益，优化生产决策是每个企业经营管理中的核心环节。某企业生产电子产品需采购两种零配件，并将其装配后形成成品，且成品合格性受零配件质量影响，因此企业需通过抽样检测控制零配件次品率，并对不合格成品选择报废或拆解。生产过程中需决策是否检测零配件或成品、是否拆解不合格品，并考虑调换市场退回品的损失。对此我们需要设计抽样方案、优化生产阶段决策，并推广到多工序多零配件场景，以通过成本与风险平衡提升整体效益。

1.2 题设数据

表一给出了两种零配件和成品的次品率，以及购买单价、检测成本、市场售价等参数信息。

表二给出了八个零配件在两道工序中的次品率、购买单价、检测成本、拆解费用等参数信息。

图一给出了给出了 2 道工序、8 个零配件的基本流程

1.3 需要解决问题

问题一：在 10% 的标称度下，求出在 95% 与 90% 的信度下使检测次数尽可能少的抽样检测方案。

问题二：基于给定的六种情况下零配件与成品的参数，给出使利润最大化的最优生产决策方案。

问题三：在 m 道工序、n 个零配件生产系统中，基于各环节的次品率、成本和售价等参数，制定最优的检测、拆解和调换决策方案，并对给定的 2 道工序、8 个零配件给出具体决策依据和量化结果。

二、 模型假设

假设一：零配件的次品率相互独立，且与成品的次品率相互独立。

假设二：厂家每轮生产策略相同，即递归过程中零配件的次品率不变。

假设三：合格成品的市场售价与不合格成品的调换损失为定值，不随市场需求变化。

假设四：单位零配件所产生的期望利润相同。

三、 符号说明

符号	说明	单位
n	样本数	-
Z_α	临界值	-
α	置信度	-
p_0	标称值	-
H_0	零假设	-
H_1	备择假设	-
X	次品数量	个
p	次品率	-
d	容忍区间	-
Π	总利润期望	元
q	合格率	-
c_j	单位零件成本	元
s	市场售价	元
x_{ij}	第 i 道工序中第 j 个部件检测状态	-
d_{ij}	第 i 道工序中第 j 个部件检测成本	个
a_j	单位成品装配成本	元
t	拆解成本	元
l	调换损失	元
θ	回收比例	-

(其余符号详见正文)

四、 问题分析

4.1 对问题一的分析

题目要求我们在标称值确定的情况下，设计出一种合理的抽样检测方案，使得在给定的两种不同情形下，抽样的次数最小。要使抽样的次数最小，即使得抽样的样本容量最小即可。基于此，我们根据经典检验样本容量公式，结合题目的标称值得到理想样本比例，同时根据不同情形下的置信度确定临界值，最终计算出抽样次数。

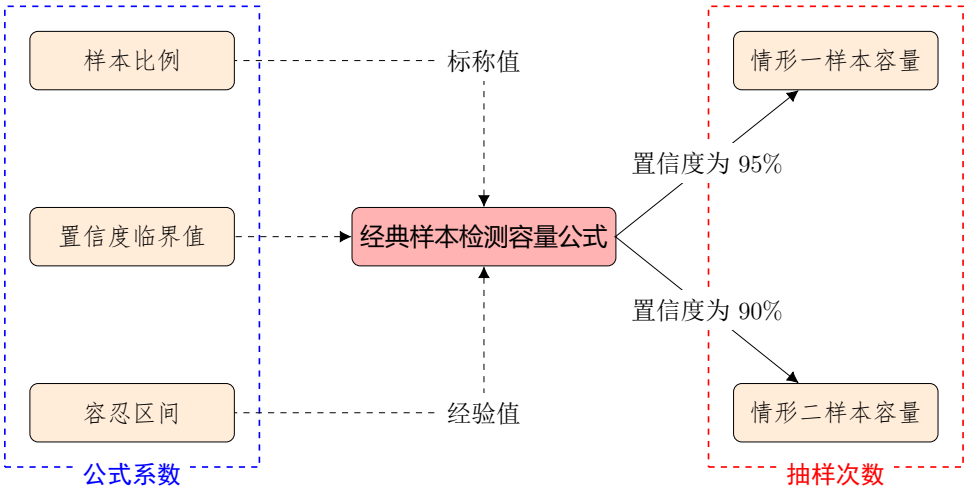


图 1　第一问流程图

4.2 对问题二的分析

题目要求我们设计出一种最优的生产决策方案，使得在给定的六种情况下，利润最大化。我们首先对各工序之间的关系，绘制了如下流程图:

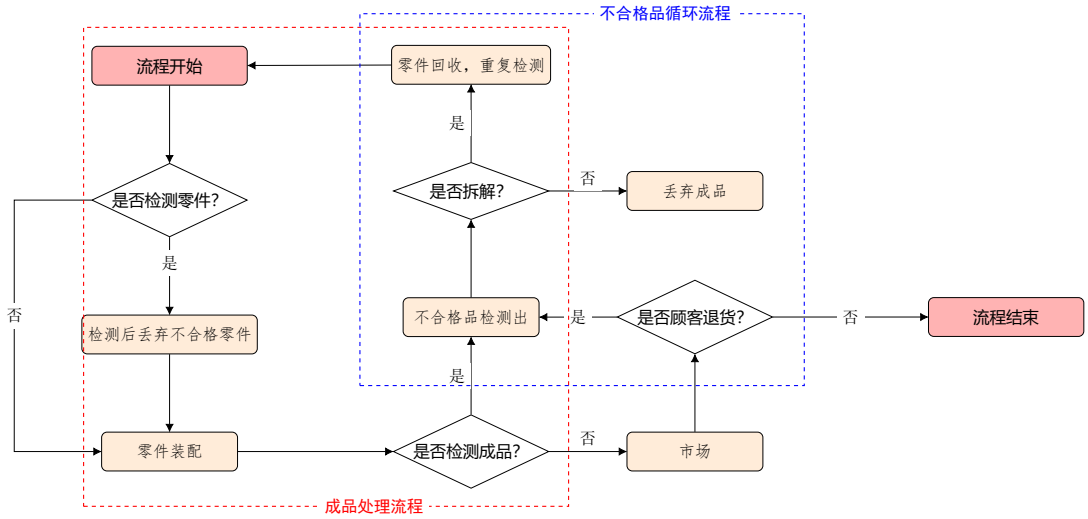


图 2　各工序关系流程图

在确定各个决策点后，我们决定利用 0-1 规划法求解最优决策方案，目标函数为期望利润，我们希望得到决策变量的取值，使得期望利润最大化。对于利润而言，考虑到拆解的零件会进入流程循环，因此该项实质上为每轮的利润的无穷求和，其中某一轮的利润可由上一轮利润乘上该轮回收的零件比例得到，显然回收比例是恒定的，故该求和的本质上为等比求和，由于回收比例小于 1，求和的结果收敛，且其数值只与回收比例和最初轮的利润相关。

问题来到如何对最初轮利润进行求解，利润的算式为收入减去成本，收入为期望收入，故为合格率与市场售价的乘积；现在对成本构成进行讨论，成本由每道工序的检测成本与固定成本、总拆解成本、总调换成本三部分组成。检测成本与拆解成本直接受检测状态与拆解状态影响，而调换成本则取决于次品率与成品检测状态。

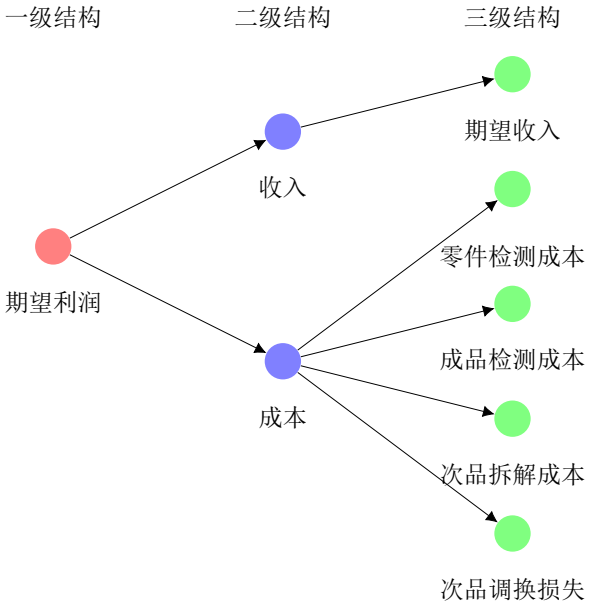


图 3　利润结构图

4.3 对问题三的分析

题目要求我们设计出一种最优的生产决策方案，使得在给定的 m 道工序、n 个零配件生产系统中，利润最大化。在问题三中，题目将 m 设定为 2，n 设定为 8，即在问题二的情况下引入半成品这一概念，对于半成品而言，同样需要进行检测以及进行拆解或者抛弃。显然的是，此问当中，利润仍然为无穷求和

五、 问题一的模型的建立和求解

5.1 模型建立

5.1.1 假设框架

根据题设条件，我们构建了如下假设框架:

- 零假设 H_0：样本比例 p 等于标称值 p_0
- 备择假设 H_1：对于拒收情形，样本比例 p 大于标称值 p_0；对于接受情形，样本比例 p 小于标称值 p_0。

5.1.2 样本容量公式

假设从总体中抽取容量为 n 的样本，记其中的次品个数为随机变量 X，显然该变量服从二项分布 $X \sim Bin(n,p)$。其中 p 为样本比例。根据中心极限定理，当 n 足够大时，X 可以近似服从正态分布，即

$$X \sim N(np, np(1-p)) \tag{1}$$

进一步的，我们可确定样本中的次品率的统计量 $\hat{p}$ 近似服从正态分布，即

$$\hat{p} \sim N(p, \frac{p(1-p)}{n}) \tag{2}$$

由上式我们可推导出经典样本检测容量公式

$$n = \frac{Z_\alpha^2 p(1-p)}{d^2} \tag{3}$$

其中 Z_α 为置信区间在标准正态分布中的临界值，α 为置信度，d 为容忍区间，一般取经验值。

5.2 模型求解

将题设条件代入上式，同时，我们取容忍区间为 0.02，最终求得结果如下表

表 1　不同置信水平下的样本量计算（$d = 0.02, p_0 = 0.10$）

置信水平	临界值 $Z_{\alpha/2}$	容忍区间 d	标称次品率 p_0	最小样本量 n
95%	1.960	0.02	0.10	865
90%	1.645	0.02	0.10	609

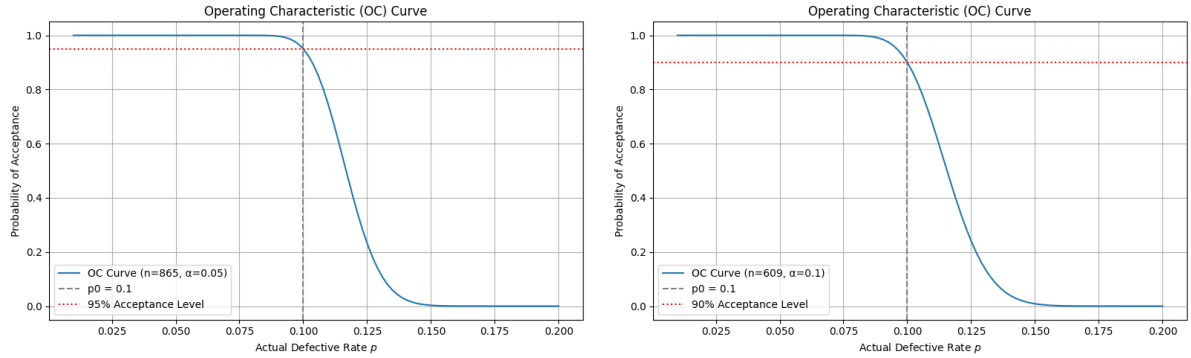
由表可知，在 95% 置信水平下，临界值 $Z_{\alpha/2} = 1.960$，容忍区间 $d = 0.02$，标称次品率 $p_0 = 0.10$，最小样本量 $n = 865$。在 90% 置信水平下，临界值 $Z_{\alpha/2} = 1.645$，容忍区间 $d = 0.02$，标称次品率 $p_0 = 0.10$，最小样本量 $n = 609$。

5.3 结果分析

上述结果表明，若零件总数远大于最小样本量时，无论样本容量为何值时，样本容量取结果值时能取到满足题设条件与统计约束上的最优值。

为进一步验证得到的样本容量的合理性，本文拟采用 OC 曲线对结果进行可视化分析，OC 曲线反映了在不同实际不合格率 p 下，产品被接受的概率 P_{accept}。

通过计算，并绘制了如下的 OC 曲线：



(a) 样本空间为 865 的 OC 曲线　　(b) 样本空间为 609 的 OC 曲线

图 4　双图

利用 OC 曲线可以直观地观察在不同实际不合格率下当前抽样计划的接受概率。从图4a中不难发现，当实际不合格率接近标称值 $p_0 = 0.10$ 时，接受概率约为 95%，与设定置信水平一致；特别地，我们注意到，随着实际不合格率上升，接受概率陡然下降，说明该检验方案对不合格产品的敏感性较高，即对不合格产品的识别能力较强。

同时，对于质量较好的批次，该方案也能保持较高的接受概率，体现了良好的灵敏度和平衡性。我们从图4b也能得到类似的结论。

此外，根据 OC 曲线我们还能得出，该种抽样方案下对不合格品的最大容忍数 $n_{\max}$，即若发现样本容量中不合格数大于该数值，则拒收，其算式为

$$n_{\max} = \frac{Z_{\alpha/2}^2 p_0(1-p_0)}{d^2} \tag{4}$$

对于情形一而言，$n_{\max} = 103$，即若抽样容量中不合格数大于 103，则拒收。对于情形二而言，$n_{\max} = 74$，即若抽样容量大于 74，则拒收。

六、 问题二的模型的建立和求解

6.1 模型建立

七、 模型的评价

7.1 模型的优点

- 优点 1：经验样本容量公式计算简便，无需复杂统计工具，并且很好的控制了两类错误，即生产方风险和使用方风险。在信度要求变化时，公式能灵活调整抽样方案，实现了动态调整。
- 优点 2：0-1 规划整合复杂逻辑约束，简要精准的描述了“是否”类二元决策问题，简化了模型，适用于多阶段决策。
- 优点 3：

7.2 模型的缺点

- 缺点 1：问题一假设样本中的次品出现是独立且概率恒定的，但实际生产中，次品可能集中在某些批次，此时抽样结果会低估真实风险。
- 缺点 2：0-1 规划的计算复杂度高，变量增多时求解时间指数级增长，增大计算难度。
- 缺点 3：

八、 模型的改进与推广

8.1 改进

- 改进 1：问题一可进行成本整合优化：将公式嵌入决策树或线性规划，同时优化样本量、检测成本、拆解费用等。
- 改进 2：

8.2 推广

- 推广 1：经验样本容量公式可推广到供应商来料检验、电子产品可靠性测试与寿命评估、市场质量反馈与售后分析等实际问题。
- 推广 2：0-1 规划可推广到投资组合优化、电路设计、广告投放、网络与路径优化等所有二元决策类问题。

参考文献

[1] 司守奎, 孙玺菁. 数学建模算法与应用[M]. 北京: 国防工业出版社, 2011.

[2] 卓金武. MATLAB 在数学建模中的应用[M]. 北京: 北京航空航天大学出版社, 2011.

[3] VÁZQUEZ J I H, GONZÁLEZ S H, VÁZQUEZ J O H, et al. Production planning through lean manufacturing and mixed integer linear programming[J]. Leather and Footwear Journal, 2021, 21(1):47-62.

[4] LEE A H, KANG H Y. A mixed 0-1 integer programming for inventory model: a case study of tft-lcd manufacturing company in taiwan[J]. Kybernetes, 2008, 37(1):66-82.

附录 A　文件列表

文件名	功能描述
exercise_1.py	问题一程序代码
exercise_2.py	问题二程序代码
exercise_3.py	问题三程序代码

附录 B　代码

exercise_1.py

```
import math
from scipy.stats import norm

# 定义参数
p0 = 0.10  # 标称次品率
alpha_95 = 0.05  # 95%置信水平
alpha_90 = 0.10  # 90%置信水平
z_95 = norm.ppf(1-alpha_95/2) # 95%的临界值
z_90 = norm.ppf(1-alpha_90/2 )  # 90%的临界值

# 计算样本量
def calculate_sample_size(z_alpha, p0, delta):
    n=(z_alpha/delta)**2*p0*(1-p0)
    return math.ceil(n)
# 假设检测误差 delta 为 5%
delta = 0.02
n_95=calculate_sample_size(z_95,p0,delta)
n_90=calculate_sample_size(z_90,p0,delta)
print(n_95,n_90)
```

exercise_2.py

```
import itertools
import pandas as pd
```

```python

# 表1数据（6种场景）
scenarios = [
    {"case":1, "p1":0.10,"c1":4,"d1":2, "p2":0.10,"c2":18,"d2":3, "pf":0.10,"cf":6,"df":3, "s":56,"L":6,"D":5},
    {"case":2, "p1":0.20,"c1":4,"d1":2, "p2":0.20,"c2":18,"d2":3, "pf":0.20,"cf":6,"df":3, "s":56,"L":6,"D":5},
    {"case":3, "p1":0.10,"c1":4,"d1":2, "p2":0.10,"c2":18,"d2":3, "pf":0.10,"cf":6,"df":3,"s":56 ,"L":30,"D":5},
    {"case":4, "p1":0.20,"c1":4,"d1":1, "p2":0.20,"c2":18,"d2":1, "pf":0.20,"cf":6,"df":2, "s":56,"L":30,"D":5},
    {"case":5, "p1":0.10,"c1":4,"d1":6, "p2":0.20,"c2":18,"d2":1, "pf":0.10,"cf":6,"df":2, "s":56,"L":10,"D":5},
    {"case":6, "p1":0.05,"c1":4,"d1":2, "p2":0.05,"c2":18,"d2":3, "pf":0.05,"cf":6,"df":3, "s":56,"L":10,"D":30},
]

# 计算利润的函数（修正版）
def calculate_profit(x1, x2, x3, x4, params):
    # 计算合格率
    q1 = 1 if x1 == 1 else (1 - params["p1"])  # 零件1合格概率
    q2 = 1 if x2 == 1 else (1 - params["p2"])  # 零件2合格概率
    qf = q1 * q2 * (1 - params["pf"])          # 成品合格概率

    # 计算各项成本和收入
    revenue = qf * params["s"]                    # 收入
    costs = (params["c1"] + params["c2"] +        # 零件成本
             x1 * params["d1"] +                  # 零件1检测成本
             x2 * params["d2"] +                  # 零件2检测成本
             params["cf"] +                       # 装配成本
             x3 * params["df"] +                  # 成品检测成本
             params["D"] * (1 - qf) * x4 +        # 拆解成本
             (1 - x3) * (1 - qf) * params["L"]) # 售后损失
```

```
30
31      # 计算单位利润
32      denominator = 1 - x4 * (1 - qf)  # 考虑循环利用的影响
33      if abs(denominator) < 1e-9:
34          return -float('inf')
35
36      return (revenue - costs) / denominator
37
38  # 遍历场景求解
39  results = []
40  for sc in scenarios:
41      max_profit = -float('inf')
42      best_decision = None
43
44      # 枚举所有0-1组合（共16种），并过滤无效组合（x4 > x3）
45      for x1, x2, x3, x4 in itertools.product([0, 1], repeat=4):
46          if x4 > x3:  # 不检测成品则不能拆解
47              continue
48
49          current_profit = calculate_profit(x1, x2, x3, x4, sc)
50
51          if current_profit > max_profit:
52              max_profit = current_profit
53              best_decision = (x1, x2, x3, x4)
54
55      # 保存结果
56      x1, x2, x3, x4 = best_decision
57      results.append({
58          "情况": sc["case"],
59          "最大单位利润": round(max_profit, 4),
60          "零配件1检测": "是" if x1 == 1 else "否",
61          "零配件2检测": "是" if x2 == 1 else "否",
62          "成品检测": "是" if x3 == 1 else "否",
63          "不合格成品拆解": "是" if x4 == 1 else "否"
64      })
```

```
65
66 # 转换为DataFrame并显示
67 df = pd.DataFrame(results)
68 # print(df.to_string(index=False))
69 from tabulate import tabulate
70
71 # 打印美化后的结果表格
72 print(tabulate(df, headers='keys', tablefmt='grid', showindex=
       False))
```

exercise_3.py

```
1
2 import itertools
3 import pandas as pd
4 import numpy as np
5
6 data = {"零配件1": {"次品率": 0.1,"购买单价": 2,"检测成本":
       1},"零配件2": {"次品率": 0.1,"购买单价": 8, "检测成本": 1,},
        "零配件3": {"次品率": 0.1, "购买单价": 12,"检测成本": 2,},"
       零配件4": { "次品率": 0.1,"购买单价": 2,"检测成本": 1, "半成
       品": {} },"零配件5": { "次品率": 0.1, "购买单价": 8,"检测成
       本": 1 },"零配件6": { "次品率": 0.1, "购买单价": 12,"检测成
       本": 2},"零配件7": { "次品率": 0.1, "购买单价": 8, "检测成本
       ": 1, },
7 "零配件8" : {"次品率": 0.1,"购买单价": 12,"检测成本": 2, },"半
       成品1" :{"次品率": 0.1,"装配费用":8,"检查成本":4,"拆解费用"
       :6},"半成品2" :{"次品率": 0.1,"装配费用":8,"检查成本":4,"拆
       解费用":6},"半成品3" :{"次品率": 0.1,"装配费用":8,"检查成本"
       :4,"拆解费用":6},"成品":{"次品率": 0.1,"装配费用":8,"检查成
       本":6,"拆解费用":10,"市场售价":200,"调换损失":30}
8 }
9
10 # 计算利润的函数
11 def calculate_profit(x11,x12,x13,x14,x15,x16,x17,x18,x21,x22,
       x23,x3,z1,z21,z22,z23, data):
```

```
    # 计算合格率
    ls1=[x11,x12,x13,x14,x15,x16,x17,x18]
    ls2=[x21,x22,x23]
    def quality(x):  # 检测返回1，不检测返回0.9
        return 1 if x == 1 else 0.9
    qf = quality(x11)*quality(x12)*quality(x13)*quality(x14)*quality(x15)*quality(x16)*quality(x17)*quality(x18)*quality(x21)*quality(x22)*quality(x23)*0.9

    p=qf*200
    S1=  sum([data[f'零配件{i}']['购买单价']+x*data[f'零配件{i}']['检测成本'] for i,x in zip(range(1,9),ls1)])
    S2=sum([data[f'半成品{i}']['装配费用']+x*data[f'半成品{i}']['检查成本'] for i,x in zip(range(1,4),ls2)])
    S3=data['成品']['装配费用']+x3*data['成品']['检查成本']
    A=sum([i*0.1*6 for i in [z21,z22,z23]])+z1*0.1*10
    pai_h_0=p-(x21*4+x22*4+x23*4)-(x3*6+8)-(1-qf)*z1* 10-(1-x3)*(1-qf)*30
    pai_h=(z1*(1-qf)*pai_h_0)/(1-z1*(1-qf))
    re=(1 - x3) * (1 - qf)*30
    E_TA=(3*z21*(1-0.9)*x21+3*z22*(1-0.9)*x22+2*z23*(1-0.9)*x23)/8
    pai_0=p-S1-S2-S3-A-re
    pai_c=pai_0/(1-E_TA)



    return pai_h+pai_c


# 遍历场景求解
results = []

max_profit = -float('inf')
best_decision = None
```

```
41
42  for x11,x12,x13,x14,x15,x16,x17,x18,x21,x22,x23,x3,z1,z21,z22,
       z23 in itertools.product([0, 1], repeat=16):
43      if z21<=x21 and z22<=x22 and z23<=x23:
44          current_profit = calculate_profit(x11,x12,x13,x14,x15,
       x16,x17,x18,x21,x22,x23,x3,z1,z21,z22,z23,data)
45          if current_profit > max_profit:
46              max_profit = current_profit
47              best_decision = (x11,x12,x13,x14,x15,x16,x17,x18,
       x21,x22,x23,x3,z1,z21,z22,z23)
48
49  # 保存结果
50  # 解包最优决策变量
51  x11, x12, x13, x14, x15, x16, x17, x18, x21, x22, x23, x3, z1,
        z21, z22, z23 = best_decision
52
53  # 构造竖直排雷表格
54  vertical_results = [
55      ("最大单位利润", round(max_profit, 4)),
56
57      # 零配件检测
58      ("零配件1检测", "是" if x11 else "否"),
59      ("零配件2检测", "是" if x12 else "否"),
60      ("零配件3检测", "是" if x13 else "否"),
61      ("零配件4检测", "是" if x14 else "否"),
62      ("零配件5检测", "是" if x15 else "否"),
63      ("零配件6检测", "是" if x16 else "否"),
64      ("零配件7检测", "是" if x17 else "否"),
65      ("零配件8检测", "是" if x18 else "否"),
66
67      # 半成品检测
68      ("半成品1检测", "是" if x21 else "否"),
69      ("半成品2检测", "是" if x22 else "否"),
70      ("半成品3检测", "是" if x23 else "否"),
71
```

```
    # 成品检测
    ("成品检测", "是" if x3 else "否"),

    # 拆解决策
    ("成品拆解", "是" if z1 else "否"),
    ("半成品1拆解", "是" if z21 else "否"),
    ("半成品2拆解", "是" if z22 else "否"),
    ("半成品3拆解", "是" if z23 else "否")
]

# 计算中间项并返回组成公式各项
def calculate_profit_components(x11,x12,x13,x14,x15,x16,x17,
   x18,x21,x22,x23,x3,z1,z21,z22,z23, data):
    ls1 = [x11,x12,x13,x14,x15,x16,x17,x18]
    ls2 = [x21,x22,x23]

    def quality(x):
        return 1 if x == 1 else 0.9

    qf = quality(x11)*quality(x12)*quality(x13)*quality(x14)*
   quality(x15)*quality(x16)*quality(x17)*quality(x18)*quality(
   x21)*quality(x22)*quality(x23)*0.9
    p = qf * 200
    S1 = sum([data[f'零配件{i}']['购买单价'] + x * data[f'零配
   件{i}']['检测成本'] for i, x in zip(range(1, 9), ls1)])
    S2 = sum([data[f'半成品{i}']['装配费用'] + x * data[f'半成
   品{i}']['检查成本'] for i, x in zip(range(1, 4), ls2)])
    S3 = data['成品']['装配费用'] + x3 * data['成品']['检查成
   本']
    A = sum([i * 0.1 * 6 for i in [z21, z22, z23]]) + z1 * 0.1
    * 10
    pai_h_0 = p - (x21 * 4 + x22 * 4 + x23 * 4) - (x3 * 6 + 8)
    - (1 - qf) * z1 * 10 - (1 - z1) * (1 - qf) * 30
    denominator = 1 - z1 * (1 - qf)
    pai_h = (z1 * (1 - qf) * pai_h_0 / denominator) if
```

```
    denominator != 0 else 0
    re = (1 - x3) * (1 - qf) * 30
    E_TA = (3 * z21 * 0.1 + 3 * z22 * 0.1 + 2 * z23 * 0.1) / 8
    denominator2 = 1 - E_TA
    pai_0 = p - S1 - S2 - S3 - A - re
    pai_c = pai_0 / denominator2 if denominator2 != 0 else 0
    total_profit = pai_h + pai_c

    return {
        "qf（合格率）": qf,
        "p（合格产品收益）": p,
        "S1（零配件成本）": S1,
        "S2（半成品成本）": S2,
        "S3（成品检测成本）": S3,
        "A（拆解成本）": A,
        "pai_h_0（补救前利润）": pai_h_0,
        "pai_h（不合格补救利润）": pai_h,
        "re（调换损失）": re,
        "E_TA（期望时间损失）": E_TA,
        "pai_0（有效利润）": pai_0,
        "pai_c（时间调整后利润）": pai_c,
        "最大单位利润（总）": total_profit
    }

# 转换为 DataFrame 并输出
df_vertical = pd.DataFrame(vertical_results, columns=["决策项", "取值"])
print(df_vertical.to_string(index=False))

# 计算详细组成项
components = calculate_profit_components(x11,x12,x13,x14,x15,x16,x17,x18,x21,x22,x23,x3,z1,z21,z22,z23, data)

# 构造中间变量结果竖直输出
print("\n□ 利润组成公式各项：\n")
```

```
131 df_components = pd.DataFrame(list(components.items()), columns
    =["计算项", "数值"])
132 print(df_components.to_string(index=False, float_format="%.4f"
    ))
```