杭州电子科技大学数学建模课程设计

承　诺　书

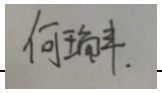
我们仔细阅读了杭州电子科技大学数学建模课程设计要求。

我们知道，抄袭别人的成果是违反数学建模课程设计要求的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守课程设计要求，以保证课程设计成绩评定的公正、公平性。如有违反课程设计要求的行为，我们将受到处理。

我们课程设计的题目：基于整数规划模型的墨盒切换模型

组队成员(打印并签名) ：

1. 姓名（打印）何瑞丰　学号 22070112　签名 

日期：2024 年 7 月 6 日

（请勿改动此页内容和格式。此承诺书打印签名后作为纸质论文的封面。以上内容请仔细核对，如填写错误，无法登记成绩。）

基于整数规划的墨盒切换策略模型

摘要

本文针对印刷机中的墨盒切换问题建立的再不考虑顺序的情况下最小化切换次数的 0-1 规划模型，和考虑墨盒顺序的最小化墨盒切换时间非线性整数规划模型，并进一步我们将非线性整数规划模型转化成线性整数规划模型，进而已知数据进行了求解。

首先，考虑印刷机**墨盒切换策略模型**，目标选择为最小换切换时间。结合问题的实际与假设，我们选择以包装种类 k 在插槽 j 上是否从墨盒 i1 切换到墨盒 i2 为决策 0-1 变量，1 表示切换，0 表示不切换。进一步分析决策变量的性质，我们推出了一些中间变量的计算方法，中间变量包括：在包装 k 时，卡槽 j 使用墨盒 i、墨盒 i 是否被使用、卡槽是否被使用使用等 0-1 中间变量。并且考虑到实际中墨盒可以在印刷机上但不被使用，因此我们又设置了墨盒 i 是否被使用为决策变量。在假设每个包装打印中每个墨盒**至多使用一次**下，通过分析这两个决策变量，我们分析得到在包装 k 时墨盒 i 在卡槽上的**顺序排名**，将墨盒不被使用定义为顺序排名为 0。这样我们便实现打印机打印包装时，保证墨盒的顺序约束要求。同时我们的假设时宽松的，如果存在墨盒在同一包装多次使用的情况，可以通过**设置虚拟墨盒**的方式满足假设要求。进一步充分考虑问题的实际，对模型的约束条件进行了完善。同时我们假设模型的初始条件为**完美条件**，即能实现目标最小的条件，我们通过定义墨盒 0 和初始步的墨盒转操作求解这一完美条件，在初始步包装打印时，所有的墨盒都需要从墨盒 0 进行切换，这一步不会产生额外的目标的消耗。最后建立完整的非线性整规划模型，为了方便后面的求解，采用了**中间变量和大 M 法**将我们的非线性整数规划模型转化成**线性整数规划模型**。

针对问题一，选择切换次数为目标并且不考虑包装打印墨盒顺序的要求，我们将我们上面建立的印刷机墨盒切换策略模型进行化简和转化，最后得到**基于 0-1 整数规划的忽略顺序最小切换次数模型**。针对给出的附件 1 中的 5 组数据，我们采用 **gurobi** 求解器进行求解，最后得到最小的切换次数依次为：**5，8，48，58，130**。部分最佳转换方案如表 2 所示，详细见附录 4 问题一的详细求解结果。

针对问题二，以最小化切换时间为目标，采用之前建立线性整数规划模型对附件 2 中的 3 个数据进行求解，得到最小切换时间依次为：**56，184，249**。部分最佳墨盒切换方案如表 4 所示，详细墨盒转化方案见附录 5 问题二的详细求解结果。

关键词　0-1 规划 线性整数规划　gurobi 求解器　非线性约束转化　印刷机墨盒切换

一、 问题的重述

柔性版印刷作为一种绿色的印刷方式，已在我国的包装行业中得到广泛应用，其采用电油墨的增材沉积方式，既减少了浪费，又实现了快速原型设计。然而，大型柔性印刷机在印刷过程中，由于插槽数量有限，无法一次性放置所有颜色的墨盒，因此需要在不同的包装印刷间进行墨盒的切换，这一切换过程需要对传墨辊和滚筒进行清洗，消耗了大量的时间，降低了印刷效率。

考虑到柔性印刷机的工作原理及其在实际生产中的应用情景，我们需要解决的问题是如何制定最优的墨盒切换策略，以最小化切换次数和切换时间，提高印刷效率。具体来说，我们需要解决的两个问题是：1）在不考虑墨盒顺序的情况下，如何设计一种数学模型，使得在给定的包装种类印刷顺序下，总切换次数最小；2）在考虑墨盒顺序并且不同颜色墨盒之间的切换时间不完全相同的情况下，如何设计一种数学模型，使得在给定的包装种类印刷顺序下，总切换时间最小。

在解决这两个问题的过程中，我们需要考虑到印刷过程中的一些实际情况，如：每种包装所需的墨盒数量和种类，插槽的数量，切换墨盒时的清洗时间等。此外，我们也需要根据已知的数据对我们的模型进行求解和验证。

二、 问题的分析

本研究的目标是寻找最优的印刷机墨盒切换策略，以最小化切换次数和切换时间。这个问题的复杂性在于，墨盒的切换不仅涉及到切换的次数，还涉及到切换的顺序。为了解决这个问题，我们将其建模为两个整数规划问题。

首先，我们考虑了不考虑顺序的情况下最小化切换次数的问题。这个问题可以看作是一个 0-1 整数规划问题，即决策变量为墨盒是否被切换，目标函数为最小化切换次数。在这个问题中，我们需要确定每个包装种类在每个插槽上的墨盒切换策略。

然后，我们考虑了考虑墨盒顺序的最小化墨盒切换时间问题。这个问题可以看作是一个非线性整数规划问题，即决策变量为墨盒的切换顺序，目标函数为最小化切换时间。然而，非线性整数规划问题在求解时可能会遇到一些困难，因此我们进一步将其转化为线性整数规划问题，以便于求解。

在建模过程中，我们引入了一些决策变量和中间变量，用于描述墨盒的使用情况和切换顺序等信息。例如，我们引入了一个决策变量，表示包装种类 k 在插槽 j 上是否从墨盒 i1 切换到墨盒 i2。我们还引入了一些中间变量，如在包装 k 时，插槽 j 使用墨盒 i 的情况，墨盒 i 是否被使用，插槽是否被使用等。

在考虑约束条件时，我们充分考虑了问题的实际情况。例如，我们考虑了每个包装打印中至多使用一次墨盒的约束，以及实际中墨盒可以在印刷机上但不被使用的情况。此外，我们还假设模型的初始条件为完美条件，即能实现目标最小的条件。

在建立完整的模型后，我们使用了 gurobi 工具对模型进行求解，得到了最优的墨盒切换策略。我们发现，通过调整墨盒的切换顺序，可以显著减少切换次数和切换时间，从而提高生产效率。

三、 模型的假设

1.假设所有的墨盒均至多使用 1 次，这一点在建立模型的时候时需要，在现实中如果一个墨盒需要使用多次，我们可以通过设置虚拟墨盒的方式实现这一点。

2.假设印刷的墨盒初始状态为完美条件，即能使得初始阶段切换早少的条件。

3.假设在决策的过程中印刷机始终能保持正常的工作，忽略意外事件的发生。
4.假设两个墨盒之间切换的时间始终为定值，不会因为印刷的次数和时间发生改变。
5.假设每个次只能清洗一个墨盒，清洗时间为总的清洗时间的和
6.假设只有一台印刷机。

四、 符号说明

符号	说明
$x_{i_1 i_2 jk}$	包装种类k在插槽j上是否从墨盒i_1切换到墨盒i_2，是为 1，否则 0
$K/I/J$	包装的总数/墨盒的总数/卡槽的总数
$W_{\text{次数}}$	总切换次数
$W_{\text{时间}}$	总的耗费的切换时间
y_{ijk}	考虑包装k时卡槽j是否为i，是为 1，否则为 0
y_{ik}^2	在包装k时墨盒i是否被使用。
y_{jk}^3	在包装k时卡槽j是否被使用。
z_{ik}	墨盒i在包装k时是否使用，0-1 变量，1 表示使用，0 表示不使用
y_{ijk}^6	表示每个实位置表示在包装k的情况下墨盒i在卡槽j上的实际使用顺序的矩阵，若为 0 则表示该卡槽并未使用对应的墨盒。
y_{ik}^7	表示在包装k时墨盒i所排卡槽上的位置排名从 1 到I_k，如果没有使用则定义为 0，

五、 模型的建立与求解

5.1 墨盒切换时间模型的建立

考虑建立非线性整数规划模型，选择决策变量$x_{i_1 i_2 jk}$为包装种类k在插槽j上是否从墨盒i_1切换到墨盒i_2，其中i_1, i_2从 0 到I，并且 0 代表空墨盒并不代指实际的墨盒编号。

定义包装编号为k总数为K，墨盒编号为i总数为I，卡槽编号为j总数为J。

总的切换数如下面的公式计算得到

$$W_{\text{次数}} = \sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i_1=0}^{I}\sum_{i_2=0}^{I} x_{i_1 i_2 jk} \tag{1}$$

定义墨盒由i_1切换到i_2所耗费的时间为$t_{i_1 i_2}$，则总的耗费的时间如下公式计算

$$W_{\text{时间}} = \sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i_1=0}^{I}\sum_{i_2=0}^{I} x_{i_1 i_2 jk} t_{i_1 i_2} \tag{2}$$

接下来，考虑约束条件，首先根据我们决策变量衍生一些看似简单但是十分有用的中间变量，考虑包装k时卡槽j是否为i，将其定义为y_{ijk}，由上面的决策变量可得y_{ijk}的计算公式如下

$$y_{i_2 jk} = \sum_{i_1=0}^{I} x_{i_1 i_2 jk} \tag{3}$$

y_{ik}^2表示在包装k时墨盒i是否在卡槽上。

$$y_{ik}^{2}=\sum_{j=1}^{J}y_{ijk} \tag{4}$$

y_{jk}^{3}表示在包装k时卡槽j是否被使用。

$$y_{jk}^{3}=\sum_{i=1}^{I}y_{ijk} \tag{5}$$

包装k时墨盒的数量如下计算

$$I_{k}=\sum_{i=1}^{I}\sum_{j=1}^{J}y_{ijk} \tag{6}$$

考虑实际印刷中可以不将墨盒拔出（调整为0）而仅仅是放在卡槽中，但是不使用，因此我们考虑增加决策变量z_{ik}表示墨盒i在包装k时是否使用，0-1 变量，1 表示使用，0 表示不使用。故公式(5)需要被进一步的修改

$$y_{jk}^{3}=\sum_{i=1}^{I}(y_{ijk}\times z_{ik}) \tag{7}$$

同时假设开始运行印刷机处于完美初始条件，即与最后的得到最优需要的初始的状态相同，我们仅需要考虑墨盒从0切换任何墨盒都不会耗费切换次数（或者时间）加入到模型中，都避免墨盒切换成0的状态。所谓完美初始条件，是能实现目标最小的条件，我们通过定义墨盒 0 和初始步的墨盒转操作求解这一完美条件，在初始步包装打印时，所有的墨盒都需要从墨盒0进行切换，这一步不会产生额外的目标的消耗。

便可实现求解这一假设。

$$y_{i_{1}0jk}=0 \tag{8}$$

接下来，我们考虑用我们的决策变量推出墨盒在卡槽上的排名，若未被使用则为0，否则从1开始到I_{k}。

首先取y_{jk}^{3}的前j_{1}项和来定义分类包装下，每个卡槽使用的顺序排名（若某个卡槽在实际生产未被使用，则它的顺序排名等于它前一个卡槽的排名）。将其定义为y_{jk}^{5}为整数变量，计算公式如下所示：

$$\begin{cases} y_{1k}^{5}=y_{1k}^{3} \\ y_{jk}^{5}=y_{(j-1)k}^{5}+y_{jk}^{3}\ \ j=2,..J \end{cases} \tag{9}$$

接下来将y_{jk}^{5}与对应的y_{ijk}再与对应的z_{ik}相乘，得到y_{ijk}^{6}表示每个实位置表示在包装k的情况下墨盒i在卡槽j上的实际使用顺序的矩阵，若为0则表示该卡槽并未使用对应的墨盒。计算公式如下所示：

$$y_{ijk}^{6}=y_{ijk}^{5}\times y_{ijk}\times z_{ik} \tag{10}$$

根据我们的假设每个包装k每个墨盒至多使用一次，如果存在一个墨盒被多次使用的情况，则采用虚拟墨盒的方式进行调整为每个墨盒只使用一次。

因此，定义y_{ik}^{7}表示在包装k时墨盒i所排卡槽上的位置排名从 1 到I_{k}，如果没有使用则定义为0，计算公式如下：

$$y_{ik}^{7}=\sum_{j=1}^{J}y_{ijk}^{6} \tag{11}$$

接下来考虑我们进行转换操作前后墨盒的状态，需要保证进行包装工作时前一步状态等于前一步的调整结束状态。

开始切换的之前的卡槽j是否使用墨盒i,计算公式为

$$y_{i_1 jk}^{pre}=\sum_{i_2=0}^{I}x_{i_1 i_2 jk} \tag{12}$$

综合上面的公式，我们完整重要的中间变量的定义，接下来结合实际和印刷生产的要求建立约束条件。

切换的前一个标签的各个墨盒的状态与切换时的各个墨盒的标签状态相同，可得下面的等式约束：

$$y_{ijk-1}=y_{ijk}^{pre}\quad k=2..K \tag{13}$$

每个卡槽至多有一个切换状态：

$$\begin{cases} y_{ijk}\leqslant 1 \\ y_{i_1 jk}^{pre}\leqslant 1 \end{cases} \tag{14}$$

设$\tilde{y}_{ik}^{*}$为第k个包装墨盒需要的使用情况，若为 1 则表示墨盒i使用，反之则表示不适用。同样地定义z_{jk}^{*}为墨盒所需要的墨盒排序情况，若为 0 则表示墨盒不被使用。得到约束如下：

$$\begin{cases} y_{ik}^{2}=y_{ik}^{*} \\ y_{ik}^{7}=z_{ik}^{*} \end{cases} \tag{15}$$

故，综合上面的公式（1-11）可得，完整的最小化切换墨盒切换时间非线性整数模型如下所示

$$\min\ W_{\text{时间}}=\sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i_1=0}^{I}\sum_{i_2=0}^{I}(x_{i_1i_2jk}\times t_{i_1i_2})$$

$$s.t.\begin{cases}y_{ijk}=\sum_{i_1=0}^{I}x_{i_1ijk}\\ y_{ik}^{2}=\sum_{j=1}^{J}y_{ijk}\\ y_{jk}^{3}=\sum_{i=1}^{I}(y_{ijk}\times z_{ik})\\ I_k=\sum_{i=1}^{I}\sum_{j=1}^{J}y_{ijk}\\ y_{i_10jk}=0\\ y_{1k}^{5}=y_{1k}^{3}\\ y_{jk}^{5}=y_{(j-1)k}^{5}+y_{jk}^{3}\ \ j=2,..J\\ y_{ijk}^{6}=y_{ijk}^{5}\times y_{ijk}\times z_{ik}\\ y_{ik}^{7}=\sum_{j=1}^{J}y_{ijk}^{6}\\ y_{i_1jk}^{pre}=\sum_{i_2=0}^{I}x_{i_1i_2jk}\\ y_{ijk-1}=y_{ijk}^{pre}\quad k=2..K\\ y_{ijk}\leqslant 1\\ y_{i_1jk}^{pre}\leqslant 1\\ y_{ik}^{2}=y_{ik}^{*}\\ y_{ik}^{7}=z_{ik}^{*}\\ x_{i_1i_2jk}\in\{0,1\}\\ z_{ik}\in\{0,1\}\\ i_1,i_2=0,1..I,j=1,2..J,k=1,2..K\end{cases}\tag{16}$$

考虑模型(15)为非线性整数规划模型，而非线性模型一般都难与求解，难找到最优解，因此接下来我们采用设置中间变量的方法和大 M 法将模型转换为线性整数规划模型我们需要处理的主要有两个等式条件分别为公式(7)y_{jk}^{3}的计算和公式(9)y_{ijk}^{6}的计算。

引入中间变量$w_{ijk}=y_{ijk}\times z_{ik}$，考虑到$y_{ijk}$和$z_{ijk}$均为 0-1 变量得到的$w_{ijk}$同样为 0-1 变量，因此我们可以很容易将$w_{ijk}$转化为线性的形式，如下所示：

$$\begin{cases}w_{ijk}\leq y_{ijk}\\ w_{ijk}\leq z_{ik}\\ w_{ijk}\geq y_{ijk}+z_{ik}-1\end{cases}\tag{17}$$

引入w_{ijk}后，公式(7)和公式(9)变化如下：

$$y_{jk}^{3}=\sum_{i=1}^{I}(y_{ijk}\times z_{ik})=\sum_{i=1}^{I}w_{ijk}$$
$$y_{ijk}^{6}=y_{ijk}^{5}\times w_{ijk} \tag{18}$$

公式(7)已经成为线性的格式，接着对y_{jk}^{6}进行线性化处理。考虑到w_{ijk}为 0-1 变量而y_{ijk}^{5}为整数变量，因此我们采用大 M 的方法将其线性化处理。

$$\begin{cases} y_{ijk}^{6}\geq y_{ijk}^{5}-M(1-w_{ijk}) \\ y_{ijk}^{6}\leq y_{ijk}^{5} \\ y_{ijk}^{6}\leq Mw_{ijk} \\ y_{ijk}^{6}\geq 0 \end{cases} \tag{19}$$

其中，M 为一较大的常数。

综合公式(15-18)得到，墨盒的线性整数规划模型

$$\min\ W_{\text{时间}}=\sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i_1=0}^{I}\sum_{i_2=0}^{I}(x_{i_1 i_2 jk}\times t_{i_1 i_2})$$

$$s.t.\begin{cases}
y_{ijk}=\sum_{i_1=0}^{I}x_{i_1 ijk}\\
y_{ik}^{2}=\sum_{j=1}^{J}y_{ijk}\\
y_{i_1 0jk}=0\\
\begin{cases} w_{ijk}\le y_{ijk}\\ w_{ijk}\le z_{ik}\\ w_{ijk}\ge y_{ijk}+z_{ik}-1\end{cases}\\
y_{jk}^{3}=\sum_{i=1}^{I}w_{ijk}\\
I_k=\sum_{i=1}^{I}\sum_{j=1}^{J}y_{ijk}\\
y_{1k}^{5}=y_{1k}^{3}\\
y_{jk}^{5}=y_{(j-1)k}^{5}+y_{jk}^{3}\ \ j=2,..J\\
\begin{cases} y_{ijk}^{6}\ge y_{ijk}^{5}-M(1-w_{ijk})\\ y_{ijk}^{6}\le y_{ijk}^{5}\\ y_{ijk}^{6}\le Mw_{ijk}\\ y_{ijk}^{6}\ge 0\end{cases}\\
y_{ik}^{7}=\sum_{j=1}^{J}y_{ijk}^{6}\\
y_{i_1 jk}^{pre}=\sum_{i_2=0}^{I}x_{i_1 i_2 jk}\\
y_{ijk-1}=y_{ijk}^{pre}\quad k=2..K\\
y_{ijk}\le 1\\
y_{i_1 jk}^{pre}\le 1\\
w_{ijk}=y_{ik}^{*}\\
y_{ik}^{7}=z_{ik}^{*}\\
x_{i_1 i_2 jk}\in\{0,1\}\\
z_{ik}\in\{0,1\}\\
i_1,i_2=0,1..I,j=1,2..J,k=1,2..K
\end{cases}\tag{20}$$

5. 2 问题一的求解

问题要求我们不考虑墨盒的顺序，并且目标为切换次数最小，根据 5.1 的总模型，可得建立的 0-1 整数规划模型如下所示

$$\min W_{\text{次数}} = \sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i_1=1}^{I}\sum_{i_2=0}^{I} x_{ijk}$$

$$s.t.\begin{cases} y_{ijk} = \sum_{i_1=0}^{I} x_{i_1 ijk} \\ y_{ik}^{2} = \sum_{j=1}^{J} y_{ijk} \\ y_{i_1 0jk} = 0 \\ y_{ik}^{7} = \sum_{j=1}^{J} y_{ijk}^{6} \\ y_{i_1 jk}^{pre} = \sum_{i_2=0}^{I} x_{i_1 i_2 jk} \\ y_{ijk-1} = y_{ijk}^{pre} \quad k = 2...K \\ y_{ijk} \leqslant 1 \\ y_{i_1 jk}^{pre} \leqslant 1 \\ y_{ik}^{2} = y_{ik}^{*} \\ x_{i_1 i_2 jk} \in \{0,1\} \\ z_{ik} \in \{0,1\} \\ i_1, i_2 = 0,1..I, j = 1,2..J, k = 1,2..K \end{cases} \tag{21}$$

针对附件 1 中的 3 个数据，采用 gurobi 对上面的 0-1 整数规划模型进行求解，得到的结果如下表所示。

表 1 附件 1 的最小切换次数的求解结果

	Ins1_5_10_2:	**Ins2_7_10_3:**	**Ins3_10_50_15:**	**Ins4_20_50_10:**	**Ins5_30_60_10:**
最小切换次数	5	8	48	58	130

部分求解具体的决策方案如下表 2 所示，详细结果见附件 3 问题 1 的详细结果。

表 2 问题一部分决策方案

Ins1_5_10_2	卡槽 1	0→5→5→4→4→7
	卡槽 2	0→7→2→2→8→3
Ins2_7_10_3	卡槽 1	0→6→6→6→6→6
	卡槽 2	0→2→8→7→7→7
	卡槽 3	0→7→3→9→9→4
Ins3_10_50_15	卡槽 1	0→17→17→4→4→4
	卡槽 2	0→35→35→29→48→48
	卡槽 3	0→14→14→14→12→12
	卡槽 4	0→11→11→11→11→11

	卡槽 5	0→7→31→43→43→43
Ins4_20_50_10	卡槽 1	0→28→28→28→28→28
	卡槽 2	0→30→30→30→30→14
	卡槽 3	0→1→1→1→7→7
	卡槽 4	0→22→22→22→22→37
	卡槽 5	0→25→25→25→25→31
Ins5_30_60_10	卡槽 1	0→44→44→3→54→10
	卡槽 2	0→11→11→11→11→11
	卡槽 3	0→56→56→56→56→56
	卡槽 4	0→5→5→38→36→36
	卡槽 5	0→55→8→8→8→34

5.3 问题二的求解

问题二考虑不同的墨盒的切换的时间时不同的，因此选择以最小化墨盒切换时间为目标函数，建立线性整数规划模型，模型如下所示

$$\min\ W_{\text{时间}}=\sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i_1=0}^{I}\sum_{i_2=0}^{I}(x_{i_1 i_2 jk}\times t_{i_1 i_2})$$

$$s.t.\begin{cases}
y_{ijk}=\sum_{i_1=0}^{I}x_{i_1 ijk}\\
y_{ik}^{2}=\sum_{j=1}^{J}y_{ijk}\\
y_{i_1 0jk}=0\\
\begin{cases} w_{ijk}\le y_{ijk}\\ w_{ijk}\le z_{ik}\\ w_{ijk}\ge y_{ijk}+z_{ik}-1\end{cases}\\
y_{jk}^{3}=\sum_{i=1}^{I}w_{ijk}\\
I_k=\sum_{i=1}^{I}\sum_{j=1}^{J}y_{ijk}\\
y_{1k}^{5}=y_{1k}^{3}\\
y_{jk}^{5}=y_{(j-1)k}^{5}+y_{jk}^{3}\ \ j=2,..J\\
\begin{cases} y_{ijk}^{6}\ge y_{ijk}^{5}-M(1-w_{ijk})\\ y_{ijk}^{6}\le y_{ijk}^{5}\\ y_{ijk}^{6}\le Mw_{ijk}\\ y_{ijk}^{6}\ge 0\end{cases}\\
y_{ik}^{7}=\sum_{j=1}^{J}y_{ijk}^{6}\\
y_{i_1 jk}^{pre}=\sum_{i_2=0}^{I}x_{i_1 i_2 jk}\\
y_{ijk-1}=y_{ijk}^{pre}\quad k=2..K\\
y_{ijk}\le 1\\
y_{i_1 jk}^{pre}\le 1\\
w_{ijk}=y_{ik}^{*}\\
y_{ik}^{7}=z_{ik}^{*}\\
x_{i_1 i_2 jk}\in\{0,1\}\\
z_{ik}\in\{0,1\}\\
i_1,i_2=0,1..I,j=1,2..J,k=1,2..K
\end{cases}\tag{22}$$

针对附件 2 的中三个数据的求解结果如下表所示，详细结果见附件 4 问题二的详细结果。

表 3 问题二数据的最小清洗时间

	Ins1_5_5_2	Ins2_10_30_10	Ins3_20_50_10
最小清洗时间（单位：分钟）	56	184	249

表 4 问题二的部分求解结果

Ins1_5_5_2	卡槽 1	0→3→2→1→3→2
	卡槽 2	0→2→1→3→1→3
Ins2_10_30_10	卡槽 1	0→17→22→22→17→17
	卡槽 2	0→29→29→29→29→11
	卡槽 3	0→6→6→6→6→6
	卡槽 4	0→26→26→17→22→22
	卡槽 5	0→23→27→27→3→29
	卡槽 6	0→5→15→15→15→15
	卡槽 7	0→25→9→11→5→5
	卡槽 8	0→8→12→21→21→21
	卡槽 9	0→28→23→8→8→26
	卡槽 10	0→22→7→7→7→7
Ins3_20_50_10	卡槽 1	0→3→47→1→1→1
	卡槽 2	0→4→6→33→33→33
	卡槽 3	0→21→30→30→30→39
	卡槽 4	0→37→35→35→20→5
	卡槽 5	0→26→36→36→36→36
	卡槽 6	0→29→49→49→49→49
	卡槽 7	0→39→39→39→41→14
	卡槽 8	0→14→45→17→17→4
	卡槽 9	0→32→42→42→7→7
	卡槽 10	0→35→9→26→8→10

六、 模型的评价与推广

6.1 墨盒切换策略模型的优点

1. 充分的结合了实际的意义，从具有实际意义的中间变量出发，推出来计算墨盒在卡槽上的顺序方式。
2. 模型具有一定的实用性和灵活性。我们的模型适用于柔性印刷机的实际工作情况，能够考虑到印刷过程中的各种约束条件，如每个包装打印中至多使用一次墨盒的约束，墨盒在印刷机上但不被使用的情况等。
3. 通过引入决策变量和中间变量，以及转化为线性整数规划问题，方便我们使用 gurobi 工具进行求解。便于找到较优解，提高了求解的速度。
4. 模型在求解过程中，同时求解了一种完美的初始条件，即怎样的初始墨盒在插槽上的状态能够实现最小的切换操作。
5. 模型假设较为宽松，没有苛刻的假设条件。

6.2 墨盒切换策略模型的缺点

1. 模型为线性整数规划模型，难于在短时间内找到最优解，只能找到一个较优的解。
2. 模型仅针对一台打印机和一台清洗机，这点导致在实际生产应用中模型的应用情景较小。

七、 参考文献

[1]乔俊伟,罗尧成.中国柔性版印刷发展现状与趋势分析[J].出版与印刷,2020,(04):60-66.DOI:10.19619/j.issn.1007-1938.2020.00.066.

[2]度巍,陈昊泽.基于 Gurobi 软件 Callback 功能的旅行商问题求解[J].电脑知识与技术,2022,18(25):9-10+25.DOI:10.14004/j.cnki.ckt.2022.1644.

[3]潘磊.非线性约束与线性约束的等价转化[D].西安电子科技大学,2019.DOI:10.27389/d.cnki.gxadu.2019.000998.

[4]徐石,李璐.技术站单组列车编组计划 0-1 规划模型研究[J].铁道运营技术,2024,30(02):9-12.DOI:10.13572/j.cnki.tdyy.2024.02.003.

[5]赵仁庆.一般指派问题的 LINGO 求解研究[J].楚雄师范学院学报,2024,39(03):73-76.

[6]刘尚一,吴涛.基于混合变量的 0-1 线性规划模型及应用[J].科学技术创新,2023,(01):92-95.

附录

附录 1

介绍：python 预处理的代码

```
K = 20  # 包装种类编号
I = 50  # 墨盒编号
J = 10  # 插槽个数
data = [[]]
matrix = [[0] * (K + 1) for _ in range(I + 1)]

input_list = [[3, 4, 21, 37, 26, 29, 39, 14, 32, 35],
              [30, 36, 49, 45, 42, 9],
              [1, 33, 30, 35, 36, 17],
              [20, 41, 7, 8],
              [14, 4],
              [8, 43, 4, 6],
              [7, 28, 5, 14],
              [17, 5, 21, 22, 6, 3, 10],
              [34, 38],
              [8, 21, 16, 46, 26, 45],
              [33, 18, 43, 6, 24, 3, 10],
              [6, 46, 28],
              [29, 19, 7, 18, 1, 21, 23, 40],
              [21, 19, 38, 9, 12],
              [15, 47, 19, 25, 4],
              [25, 12, 14, 23, 15],
              [10, 35, 12, 45],
              [33, 34, 10, 23],
              [35, 29, 45, 28, 43, 8, 24],
              [14, 49, 16, 10, 1, 41]]
for ii in range(len(input_list)):
    i = 1
    hang = input_list[ii]
    for col in hang:
        # print(row_index)
        # matrix[col][ii + 1] = 1
        matrix[col][ii + 1] = i
        i = i + 1

```

附录 2

介绍：问题一的求解代码

```
import gurobipy as gp
from gurobipy import GRB
```

```
    I = 10 #墨盒的总数
    K = 5 #包装的总数
    J = 3 #插槽的总数
    spend = [[0 if i == j else 1 for j in range(I+1)] for i in rang
e(I+1)]
    # y_2[i,k]
    y_2 = [
        [0,0, 0, 0, 0, 0],
        [0,0, 0, 0, 0, 0],
        [0,0, 1, 1, 0, 0],
        [0,0, 0, 0, 0, 1],
        [0,0, 0, 1, 0, 0],
        [0,1, 1, 0, 0, 0],
        [0,0, 0, 0, 0, 0],
        [0,1, 0, 0, 0, 1],
        [0,0, 0, 0, 1, 0],
        [0,0, 0, 0, 0, 0],
        [0,0, 0, 0, 0, 0]
    ]

    # #实际要求各个墨盒的前后顺序
    # ZO = [[],[],[]]

    # 创建模型
    model = gp.Model("mohefenpei")

    #创建变量
    x = model.addVars(range(I+1),range(I+1),range(1, J+1), range(1,
 K+1), vtype=GRB.BINARY, name="x")
    # 把所有的0 的情况都剔除
    #包装k 时卡槽j 是否为i 0-1
    y1 = model.addVars(range(I+1),range(1, J+1), range(1, K+1), vty
pe=GRB.BINARY, name="y1")
    # y1_zero = model.addVars(range(1,I+1),range(1, J+1), range(1,
K+1), vtype=GRB.BINARY, name="y1")

    # 表示包装k 是否使用墨盒i 0-1
    y2 = model.addVars(range(1,I+1), range(1,K+1), vtype=GRB.BINARY
, name="y2")

    # # 表示包装k 时卡槽j 是否被使用
    # y3 = model.addVars(range(1, J+1), range(1, K+1), vtype=GRB.BI
NARY, name="y3")
```

```
    # 表示转换前，墨盒是否处于使用状态 k 从 1 开始，k =1 y_pre = 0
    y_pre = model.addVars(range(I+1),range(1, J+1), range(1,K+1), vtype=GRB.BINARY, name="y_pre")

    # # 表示墨盒的顺序
    # z = model.addVars(range(I+1), range(1, K+1), vtype=GRB.INTEGER, name="z")
    #
    # 表示包装 k 是否使用墨盒 i    3###是否处于使用状态
    z2 = model.addVars(range(1,I+1), range(1,K+1), vtype=GRB.BINARY, name="z2")
    #设置目标函数
    obj = gp.quicksum(x[i1,i2, j, k] * spend[i1][i2] for i1 in range(I+1) for i2 in range(1,I+1) for j in range(1,J+1) for k in range(1,K+1))
    model.setObjective(obj, GRB.MINIMIZE)
    #设置约束条件
    #
    # 添加约束：y1_{i2jk} = sum_{i1=0}^I x_{i1i2jk}
    #计算y1 y_pre

    for i2 in range(I+1):
        for j in range(1,J+1):
            for k in range(1,K+1):
                model.addConstr(y1[i2, j, k] == sum(x[i1, i2, j, k] for i1 in range(I+1)), name=f"constraint_{i2}_{j}_{k}")#

    for i1 in range(I+1):
        for j in range(1,J+1):
            for k in range(1,K+1):
                model.addConstr(y_pre[i1, j, k] == sum(x[i1, i2, j, k] for i2 in range(I+1)), name=f"constraint_{i1}_{j}_{k}")#


    # # 保证包装 1 的时候所有的都是从 0 转化来的
    # for i in range(I+1):
    #     for j in range(1,J+1):
    #         model.addConstr(y_pre[i, j, 1] == 0,name=f"constraint_chushitiaojian")
    # 保证包装 1 的时候所有的都是从 0 转化来的
    for i1 in range(1,I+1):
        for i2 in range(I + 1):
            for j in range(1,J+1):
```

```
                model.addConstr(x[i1,i2, j, 1] == 0,name=f"constrai
nt_chushitiaojian")
    # 禁止替换成0
    for i1 in range(I+1):
        for j in range(1,J+1):
            for k in range(1,K+1):
                model.addConstr(x[i1,0, j, k] == 0,name=f"constrain
t_chushitiaojian")


    # #计算 y3
    # for j in range(1, J+1):
    #     for k in range(1, K+1):
    #         model.addConstr(y3[j, k] == sum(y1[i, j, k] for i in
range(1, I+1)), name=f"constraint_y_3_{j}_{k}")

    #计算y2
    for i in range(1,I+1):
        for k in range(1, K+1):
            model.addConstr(y2[i,k] == sum(y1[i, j, k] for j in ran
ge(1, J+1)), name=f"constraint_y_2_{j}_{k}")

    # 符合实际采用墨盒相同。
    # for i in range(1,I+1):
    #     for k in range(1, K+1):
    #         model.addConstr(y2[i,k] == y_2[i][k], name=f"constrai
nt_fuheyaoqiu_{i}_{k}")
    #线性化
    for i in range(1,I+1):
        for k in range(1, K+1):
            model.addConstr(z2[i,k] >= y_2[i][k], name=f"constraint
_fuheyaoqiu1_{i}_{k}")
            model.addConstr(y2[i,k] >= y_2[i][k], name=f"constraint
_fuheyaoqiu2_{i}_{k}")
            model.addConstr(z2[i,k] + y2[i,k] - 1 <= y_2[i][k], nam
e=f"constraint_fuheyaoqiu3_{i}_{k}")


    #前后一致转化一致
    for k in range(2,K+1):
        for i in range(0,I+1):
```

```
            for j in range(1, J+1):
                model.addConstr(y1[i,j,k-1] == y_pre[i,j,k], name=f"constraint_fuheyaoqiu_{i}_{k}")

    # #计算 y3
    # for j in range(1, J+1):
    #     for k in range(1, K+1):
    #         model.addConstr(y3[j, k] == sum(y1[i, j, k] for i in range(1, I+1)), name=f"constraint_y_3_{j}_{k}")
    #
    # # 计算z
    # for k in range(1, K+1):
    #     for i in range(1, I+1):
    #         model.addConstr(z[i,k] == sum(y1[i, j, k]*(sum(y3[j1, k] for j1 in range(j))) for j in range(1, J+1)), name=f"constraint_z{i}_{k}")
    #
    # for k in range(1, K+1):
    #     for i in range(1, I+1):
    #         model.addConstr(z[i,k] == z[i][k], name=f"constraint_z_3_{i}_{k}")
    #
    #
    model.setParam('TimeLimit', 3600) #时间限制，秒为单位
    # model.setParam('MIPGap', 0.1) #间隔限制
    # model.setParam('OptimalityTol', 1e-6) #公差限制

    model.setParam('IterationLimit', 15000 ) #迭代次数限制


    # 求解模型
    model.optimize()

    if model.status == GRB.OPTIMAL:
        print("Optimal objective value: ", model.objVal)


        print("\nValues of decision variables x:")
        for i1 in range(I + 1):
            for i2 in range(I + 1):
                for j in range(1, J + 1):
                    for k in range(1, K + 1):
                        if x[i1, i2, j, k].X > 0:
```

```
                            print(f"x[{i1}, {i2}, {j}, {k}] = {x[i1
, i2, j, k].X}")


        print("\nValues of decision variables y1:")
        for i in range(I + 1):
            for j in range(1, J + 1):
                for k in range(1, K + 1):
                    if y1[i, j, k].X > 0:
                        print(f"y1[{i}, {j}, {k}] = {y1[i, j, k].X}
")


        print("\nValues of decision variables y2:")
        for i in range(1, I + 1):
            for k in range(1, K + 1):
                if y2[i, k].X > 0:
                    print(f"y2[{i}, {k}] = {y2[i, k].X}")


        print("\nValues of decision variables y_pre:")
        for i in range(I + 1):
            for j in range(1, J + 1):
                for k in range(1, K + 1):
                    if y_pre[i, j, k].X > 0:
                        print(f"y_pre[{i}, {j}, {k}] = {y_pre[i, j,
 k].X}")
    else:
        print("No optimal solution found.")

    # 定义一个函数，用于将结果和解释写入文件
    def write_results_to_file(filename):
        with open(filename, 'w', encoding='utf-8') as file:
            if model.status == GRB.OPTIMAL:
                file.write("最优目标值: {}\n".format(model.objVal))

                # 打印每个卡槽上的墨盒切换状态
                file.write("\n 各个卡槽 j 上墨盒 i 的切换状态:\n")
                file.write("x[i1, i2, j, k]: 在第 k 个包装中，第 j 个插
槽从墨盒 i1 更换到墨盒 i2。\n")
                for k in range(1, K + 1):
                    file.write(f"\n 第{k}个包装的墨盒切换情况:\n")
                    for j in range(1, J + 1):
                        file.write(f"  卡槽{j}:\n")
```

```
                        for i1 in range(I + 1):
                            for i2 in range(I + 1):
                                if x[i1, i2, j, k].X > 0:
                                    file.write(f"    从墨盒{i1}切换到墨盒{i2}\n")
            else:
                file.write("未找到最优解。\n")

    # 调用函数，保存结果到指定文件
    write_results_to_file('wen1_results.txt')
```

附录 3

介绍：问题二的求解代码

```
    import gurobipy as gp
    from gurobipy import GRB

    #创建变量
    x = model.addVars(range(I+1),range(I+1),range(1, J+1), range(1, K+1), vtype=GRB.BINARY, name="x")
    # 把所有的0 的情况都剔除
    #包装k 时卡槽j 是否为i 0-1
    y1 = model.addVars(range(I+1),range(1, J+1), range(1, K+1), vtype=GRB.BINARY, name="y1")
    # y1_zero = model.addVars(range(1,I+1),range(1, J+1), range(1, K+1), vtype=GRB.BINARY, name="y1")

    # 表示包装k 是否使用墨盒i 0-1
    y2 = model.addVars(range(1,I+1), range(1,K+1), vtype=GRB.BINARY, name="y2")

    # # 表示包装k 时卡槽j 是否被使用
    y3 = model.addVars(range(1, J+1), range(1, K+1), vtype=GRB.BINARY, name="y3")

    # 表示转换前，墨盒是否处于使用状态 k 从1 开始，k =1 y_pre = 0
    y_pre = model.addVars(range(I+1),range(1, J+1), range(1,K+1), vtype=GRB.BINARY, name="y_pre")

    # 表示墨盒的顺序
    z = model.addVars(range(I+1), range(1, K+1), vtype=GRB.INTEGER, name="z")
    # 表示前j 列墨盒使用的排次
```

```
    y4 = model.addVars(range(I+1), range(1, K+1), vtype=GRB.INTEGER, name="y4")

    y5 = model.addVars(range(J+1), range(1, K+1), vtype=GRB.INTEGER, name="y5")

    y6 = model.addVars(range(I+1),range(J+1), range(1, K+1), vtype=GRB.INTEGER, name="y6")

    y7 = model.addVars(range(I+1), range(1, K+1), vtype=GRB.INTEGER, name="y7")



    # 表示包装k是否使用墨盒i    3###是否处于使用状态
    z2 = model.addVars(range(1,I+1), range(1,K+1), vtype=GRB.BINARY, name="z2")

    #设置目标函数
    obj = gp.quicksum(x[i1,i2, j, k] * int(spend[i1][i2]) for i1 in range(I+1) for i2 in range(1,I+1) for j in range(1,J+1) for k in range(1,K+1))
    model.setObjective(obj, GRB.MINIMIZE)
    #设置约束条件
    #

    # 添加约束：y1_{i2jk} = sum_{i1=0}^I x_{i1i2jk}
    #计算y1 y_pre
    for i2 in range(I+1):
        for j in range(1,J+1):
            for k in range(1,K+1):
                model.addConstr(y1[i2, j, k] == sum(x[i1, i2, j, k] for i1 in range(I+1)), name=f"constraint_{i2}_{j}_{k}")#

    for i1 in range(I+1):
        for j in range(1,J+1):
            for k in range(1,K+1):
                model.addConstr(y_pre[i1, j, k] == sum(x[i1, i2, j, k] for i2 in range(I+1)), name=f"constraint_{i1}_{j}_{k}")#


    # # 保证包装1的时候所有的都是从0转化来的
    # for i in range(I+1):
    #     for j in range(1,J+1):
```

```
58.    #         model.addConstr(y_pre[i, j, 1] == 0,name=f"constraint
_chushitiaojian")
59.    # 保证包装1的时候所有的都是从0转化来的
60.    # for i1 in range(1,I+1):
61.    #     for i2 in range(I + 1):
62.    #         for j in range(1,J+1):
63.    #             model.addConstr(x[i1,i2, j, 1] == 0,name=f"constr
aint_chushitiaojian")
64.    for i1 in range(1,I+1):
65.        for i2 in range(I + 1):
66.            for j in range(1,J+1):
67.                model.addConstr(x[i1,i2, j, 1] == 0,name=f"constrai
nt_chushitiaojian")
68.
69.
70.    # 禁止替换成0
71.    for i1 in range(I+1):
72.        for j in range(1,J+1):
73.            for k in range(1,K+1):
74.                model.addConstr(x[i1,0, j, k] == 0,name=f"constrain
t_chushitiaojian")
75.
76.    # #计算 y3
77.    # for j in range(1, J+1):
78.    #     for k in range(1, K+1):
79.    #         model.addConstr(y3[j, k] == sum(y1[i, j, k] for i in
range(1, I+1)), name=f"constraint_y_3_{j}_{k}")
80.
81.    #计算y2
82.    for i in range(1,I+1):
83.        for k in range(1, K+1):
84.            model.addConstr(y2[i,k] == sum(y1[i, j, k] for j in ran
ge(1, J+1)), name=f"constraint_y_2_{j}_{k}")
85.
86.    # 符合实际采用墨盒相同。
87.    # 未线性化的模型
88.    # for i in range(1,I+1):
89.    #     for k in range(1, K+1):
90.    #         model.addConstr(z2[i,k]*y2[i,k] == y_2[i][k], name=f"
constraint_fuheyaoqiu_{i}_{k}")
91.
92.    #线性化
93.    for i in range(1,I+1):
94.        for k in range(1, K+1):
```

```
            model.addConstr(z2[i,k] >= y_2[i][k], name=f"constraint_fuheyaoqiu1_{i}_{k}")
            model.addConstr(y2[i,k] >= y_2[i][k], name=f"constraint_fuheyaoqiu2_{i}_{k}")
            model.addConstr(z2[i,k] + y2[i,k] - 1 <= y_2[i][k], name=f"constraint_fuheyaoqiu3_{i}_{k}")


    #前后一致转化一致
    for k in range(2,K+1):
        for i in range(0,I+1):
            for j in range(1, J+1):
                model.addConstr(y1[i,j,k-1] == y_pre[i,j,k], name=f"constraint_fuheyaoqiu_{i}_{k}")

    #计算 y3
    for j in range(1, J+1):
        for k in range(1, K+1):
            model.addConstr(y3[j, k] == sum(y1[i, j, k] * z2[i,k] for i in range(1, I+1)), name=f"constraint_y_3_{j}_{k}")

    # y3 线性化
    # w[i,j,k] = y1[i, j, k] * z2[i,k]   纯纯0-1
    w = model.addVars(range(I+1),range(1, J+1), range(1,K+1), vtype=GRB.BINARY, name="w")
    for k in range(1,K+1):
        for j in range(1,J+1):
            for i in range(1,I+1):
                # model.addConstr(w[i, j, k] == y1[i, j, k] * z2[i, k], name=f"y6jisuan_{i}_{j}_{k}")
                model.addConstr(w[i, j, k] <= y1[i, j, k] , name=f"y3_1_{i}_{j}_{k}")
                model.addConstr(w[i, j, k] <= z2[i,k], name=f"y3_2_{i}_{j}_{k}")
                model.addConstr(w[i, j, k] >= z2[i,k] + y1[i,j,k] - 1, name=f"y3_2_{i}_{j}_{k}")


    # 计算y4
    # for k in range(1, K+1):
    #     for i in range(1, I+1):
```

```
#         model.addConstr(y4[i,k] == (sum(y3[j1,k]*z2[i,k] for j1 in range(1,j+1))))
# # 上面的线性化，引入中间变量 w(ik)

#计算 y5  整数 不再是0-1
for k in range(1,K+1):
    model.addConstr(y5[1,k] == y3[1,k], name=f"cumsum_0_{k}")

for k in range(1,K+1):
    for j in range(2,J+1):
        # y5[j, k] 应等于 y5[j-1, k] + y3[j, k]
        model.addConstr(y5[j,k] == y5[j-1, k] + y3[j, k], name=f"cumsum_{j}_{k}")


# #计算y6
# for k in range(1,K+1):
#     for j in range(1,J+1):
#         for i in range(1,I+1):
#             model.addConstr(y6[i,j, k] == y5[j,k] * y1[i,j,k] * z2[i,k], name=f"y6jisuan_{i}_{j}_{k}")

# w[i,j,k] = y1[i,j,k] * z2[i,k]
# w = model.addVars(range(I+1),range(1, J+1), range(1,K+1), vtype=GRB.BINARY, name="w")
# for k in range(1,K+1):
#     for j in range(1,J+1):
#         for i in range(1,I+1):
#             model.addConstr(w[i, j, k] == y1[i, j, k] * z2[i, k], name=f"y6jisuan_{i}_{j}_{k}")
#


# for k in range(1,K+1):
#     for j in range(1,J+1):
#         for i in range(1,I+1):
#             model.addConstr(y6[i,j, k] == y5[j,k] * w[i,j,k], name=f"y6jisuan_{i}_{j}_{k}")
#


#线性化 y5 是整数，而w 是0-1，这时采用大M 法进行线性化处理 M 是一个很大的数
M = 100
```

```
165.  for k in range(1,K+1):
166.      for j in range(1,J+1):
167.          for i in range(1,I+1):
168.              model.addConstr(y6[i,j,k] <= y5[j,k], name=f"y6_1ji
suan_{i}_{j}_{k}")
169.              model.addConstr(y6[i, j, k] >=0, name=f"y6_2jisuan_
{i}_{j}_{k}")
170.              model.addConstr(y6[i, j, k] >= y5[j,k]-M*(1-
w[i,j,k]), name=f"y6_3jisuan_{i}_{j}_{k}")
171.              model.addConstr(y6[i, j, k] <= M*w[i,j,k], name=f"y
6_4jisuan_{i}_{j}_{k}")
172.
173.
174.  #计算y7
175.  for k in range(1,K+1):
176.      for i in range(1,I+1):
177.          model.addConstr(y7[i,k] == sum(y6[i,j,k] for j in range
(1,J+1)), name=f"y7jisuan_{i}_{j}_{k}")
178.
179.  for k in range(1,K+1):
180.      for i in range(1,I+1):
181.          model.addConstr(y7[i,k] == Z0[i][k] ,name=f"y7jisuan_{i
}_{j}_{k}")
182.
183.
184.  # w = {}
185.  #
186.  # w = model.addVars(range(I+1),range(1, J+1), range(1,K+1), vty
pe=GRB.BINARY, name="w")
187.  # for k in range(1, K+1):
188.  #     for i in range(1, I+1):
189.  #         y4[i, k] = model.addVar(vtype=GRB.INTEGER, name=f"y4_
{i}_{k}")
190.  #         for j1 in range(1, j+1):
191.  #             # w[i, j1, k] = model.addVar(vtype=GRB.BINARY, na
me=f"w_{i}_{j1}_{k}")
192.  #             model.addConstr(w[i, j1, k] <= y3[j1, k], name=f"
w_le_y3_{i}_{j1}_{k}")
193.  #             model.addConstr(w[i, j1, k] <= z2[i, k], name=f"w
_le_z2_{i}_{k}")
194.  #             model.addConstr(w[i, j1, k] >= y3[j1, k] + z2[i,
k] - 1, name=f"w_ge_y3z2_{i}_{j1}_{k}")
195.  #
```

```
#         model.addConstr(y4[i, k] == sum(w[i, j1, k] for j1 in range(1, j+1)), name=f"constraint_y4_{i}_{k}")
#
#
# w1 = model.addVars(range(I+1),range(1, J+1), range(1,K+1), vtype=GRB.BINARY, name="w")
# # 引入辅助变量把z的计算过程线性化
#
# for k in range(1, K+1):
#     for i in range(1, I+1):
#         z[i, k] = model.addVar(vtype=GRB.INTEGER, name=f"z_{i}_{k}")
#         for j in range(1, J+1):
#             w1[i, j, k] = model.addVar(vtype=GRB.INTEGER, name=f"w1_{i}_{j}_{k}")
#             model.addConstr(w1[i, j, k] <= y1[i, j, k], name=f"w1_le_y1_{i}_{j}_{k}")
#             model.addConstr(w1[i, j, k] <= y4[i, k], name=f"w1_le_y4_{i}_{k}")
#             model.addConstr(w1[i, j, k] >= y1[i, j, k] + y4[i, k] - 1, name=f"w_ge_y1y4_{i}_{j}_{k}")
#         model.addConstr(z[i, k] == sum(w1[i, j, k] for j in range(1, J+1)), name=f"constraint_z{i}_{k}")
#
# for k in range(1, K+1):
#     for i in range(1, I+1):
#         model.addConstr(z[i,k] == sum(y1[i, j, k]  * y4[i,k] for j in range(1, J+1)), name=f"constraint_z{i}_{k}")
#


# # 计算z
# for k in range(1, K+1):
#     for i in range(1, I+1):
#         model.addConstr(z[i,k] == sum(y1[i, j, k]  * (sum(y3[j1,k] * z2[i,k] for j1 in range(1,j+1))) for j in range(1, J+1)), name=f"constraint_z{i}_{k}")

# w = model.addVars(range(I+1),range(1, J+1), range(1,K+1), vtype=GRB.BINARY, name="w")
# u = model.addVars(range(I+1),range(1, J+1), range(1, J+1),range(1,K+1), vtype=GRB.BINARY, name="u")
#


```

```
227.  # for k in range(1, K+1):
228.  #     for i in range(1, I+1):
229.  #         for j in range(1, J+1):
230.  #             #w[i, j, k] = model.addVar(vtype=GRB.INTEGER, nam
e=f"w_{i}_{j}_{k}")
231.  #             model.addConstr(w[i, j, k] == sum(u[i, j, j1, k]
for j1 in range(1, j+1)),
232.  #                             name=f"def_w_{i}_{j}_{k}")
233.  #             for j1 in range(1, j+1):
234.  #                 #u[i, j, j1, k] = model.addVar(vtype=GRB.INTE
GER, name=f"u_{i}_{j}_{j1}_{k}")
235.  #                 model.addConstr(u[i, j, j1, k] <= y3[j1, k],
name=f"u_le_y3_{i}_{j}_{j1}_{k}")
236.  #                 model.addConstr(u[i, j, j1, k] <= z2[i, k], n
ame=f"u_le_z2_{i}_{j}_{k}")
237.  #                 model.addConstr(u[i, j, j1, k] >= y3[j1, k] +
 z2[i, k] - 1, name=f"u_ge_y3z2_{i}_{j}_{j1}_{k}")
238.  #
239.  # model.addConstrs((z[i, k] == sum(y1[i, j, k] * w[i, j, k] for
 j in range(1, J+1)) for i in range(1, I+1) for k in range(1, K+1)),
240.  #                  name="constraint_z")
241.
242.  for k in range(1, K+1):
243.      for i in range(1, I+1):
244.          model.addConstr(z[i,k] == Z0[i][k], name=f"constraint_z
_3_{i}_{k}")
245.
246.  model.setParam('TimeLimit', 3600) #时间限制，秒为单位
247.  # model.setParam('MIPGap', 0.1) #间隔限制
248.  # model.setParam('OptimalityTol', 1e-6) #公差限制
249.
250.  model.setParam('IterationLimit', 2000000 ) #迭代次数限制
251.  # 求解模型
252.  model.optimize()
253.
254.  if model.status == GRB.OPTIMAL:
255.      print("Optimal objective value: ", model.objVal)
256.
257.
258.      print("\nValues of decision variables x:")
259.      for i1 in range(I + 1):
260.          for i2 in range(I + 1):
261.              for j in range(1, J + 1):
262.                  for k in range(1, K + 1):
```

```
                        if x[i1, i2, j, k].X > 0:
                            print(f"x[{i1}, {i2}, {j}, {k}] = {x[i1
, i2, j, k].X}")


    print("\nValues of decision variables y1:")
    for i in range(I + 1):
        for j in range(1, J + 1):
            for k in range(1, K + 1):
                if y1[i, j, k].X > 0:
                    print(f"y1[{i}, {j}, {k}] = {y1[i, j, k].X}
")


    print("\nValues of decision variables y2:")
    for i in range(1, I + 1):
        for k in range(1, K + 1):
            if y2[i, k].X > 0:
                print(f"y2[{i}, {k}] = {y2[i, k].X}")


    print("\nValues of decision variables y_pre:")
    for i in range(I + 1):
        for j in range(1, J + 1):
            for k in range(1, K + 1):
                if y_pre[i, j, k].X > 0:
                    print(f"y_pre[{i}, {j}, {k}] = {y_pre[i, j,
 k].X}")
else:
    print("No optimal solution found.")

def write_results_to_file(filename):
    with open(filename, 'w', encoding='utf-8') as file:
        if model.status == GRB.OPTIMAL:
            file.write("最优目标值: {}\n".format(model.objVal))

            # 打印每个卡槽上的墨盒切换状态
            file.write("\n 各个卡槽 j 上墨盒 i 的切换状态:\n")
            file.write("x[i1, i2, j, k]: 在第 k 个包装中，第 j 个插
槽从墨盒 i1 更换到墨盒 i2。\n")
            for k in range(1, K + 1):
                file.write(f"\n 第{k}个包装的墨盒切换情况:\n")
                for j in range(1, J + 1):
                    file.write(f"  卡槽{j}:\n")
```

```
                        for i1 in range(I + 1):
                            for i2 in range(I + 1):
                                if x[i1, i2, j, k].X > 0:
                                    file.write(f"    从墨盒{i1}切换到墨盒{i2}\n")
            else:
                file.write("未找到最优解。\n")

    # 调用函数，保存结果到指定文件
    write_results_to_file('wen2_result3.txt')
```

附录 4

介绍：问题一的详细求解结果

- **Ins1_5_10_2:**

  最优目标值: 5.0
  卡槽 1 的墨盒变化情况:
  0→5→5→4→4→7
  卡槽 2 的墨盒变化情况:
  0→7→2→2→8→3

- **Ins2_7_10_3:**

  最优目标值: 8.0
  卡槽 1 的墨盒变化情况:
  0→6→6→6→6→6→6→1
  卡槽 2 的墨盒变化情况:
  0→2→8→7→7→7→7→7
  卡槽 3 的墨盒变化情况:
  0→7→3→9→9→4→2→3

- **Ins3_10_50_15:**

  最优目标值: 48.0
  卡槽 1 的墨盒变化情况:
  0→17→17→4→4→4→4→4→4→4→4
  卡槽 2 的墨盒变化情况:
  0→35→35→29→48→48→48→48→18→18→38
  卡槽 3 的墨盒变化情况:
  0→14→14→14→12→12→47→47→26→26→26
  卡槽 4 的墨盒变化情况:
  0→11→11→11→11→11→11→28→28→28→28
  卡槽 5 的墨盒变化情况:
  0→7→31→43→43→43→43→43→33→41→41
  卡槽 6 的墨盒变化情况:
  0→24→24→30→30→30→30→30→40→20→20
  卡槽 7 的墨盒变化情况:

0→10→10→10→10→10→10→13→36→8→24
卡槽 8 的墨盒变化情况:
0→23→23→23→23→38→42→7→7→3→3
卡槽 9 的墨盒变化情况:
0→21→21→1→25→25→25→25→19→47→9
卡槽 10 的墨盒变化情况:
0→40→40→27→27→27→27→27→27→5→2
卡槽 11 的墨盒变化情况:
0→15→15→15→31→31→31→31→23→39→39
卡槽 12 的墨盒变化情况:
0→6→6→6→6→6→6→6→6→6→6
卡槽 13 的墨盒变化情况:
0→16→16→16→16→16→16→16→16→12→12
卡槽 14 的墨盒变化情况:
0→42→42→19→19→41→41→22→15→1→1
卡槽 15 的墨盒变化情况:
0→33→33→8→44→26→5→17→49→34→34

- **Ins4_20_50_10:**

最优目标值: 58.0
卡槽 1 的墨盒变化情况:
0→28→28→28→28→28→28→28→28→28→28→30→30→25→25→17→17→17→17→15→15
卡槽 2 的墨盒变化情况:
0→30→30→30→30→14→14→14→15→18→18→18→23→23→23→23→23→23→23→23→23
卡槽 3 的墨盒变化情况:
0→1→1→1→7→7→34→34→34→2→2→2→40→34→7→7→7→36→12→12→48
卡槽 4 的墨盒变化情况:
0→22→22→22→22→37→37→21→21→21→21→21→21→21→21→18→9→40→22→22→22
卡槽 5 的墨盒变化情况:
0→25→25→25→25→31→31→31→31→31→31→8→8→16→28→28→28→28→28→28→28
卡槽 6 的墨盒变化情况:
0→11→11→11→11→11→11→11→11→26→4→4→4→2→45→45→45→49→44→44→5
卡槽 7 的墨盒变化情况:
0→3→3→3→3→3→3→3→3→3→3→3→3→3→3→3→3→33→33→4→4
卡槽 8 的墨盒变化情况:
0→40→40→40→40→40→44→44→44→20→20→10→10→48→41→41→10→10→6→6→35
卡槽 9 的墨盒变化情况:

0→36→36→36→12→46→46→46→27→27→42→32→32→20→20→20→20→20→20→20→20

卡槽 10 的墨盒变化情况:

0→2→2→41→13→38→38→35→35→35→35→5→5→39→43→43→43→43→43→13→13

- **Ins5_30_60_10:**

最优目标值: 130.0

卡槽 1 的墨盒变化情况:

0→44→44→3→54→10→10→5→5→22→22→22→46→46→3→3→3→3→3→26→37→37→16→43→32→35→35→35→36→33→33

卡槽 2 的墨盒变化情况:

0→11→11→11→11→11→11→11→14→11→16→16→16→59→15→15→15→15→15→45→24→24→24→24→24→21→41→41→41→41→30

卡槽 3 的墨盒变化情况:

0→56→56→56→56→56→56→56→37→38→38→38→38→38→37→22→14→14→31→42→54→35→25→25→25→54→54→34→47→44→44

卡槽 4 的墨盒变化情况:

0→5→5→38→36→36→36→36→1→32→52→52→52→34→34→23→33→29→29→29→20→20→3→3→3→3→3→3→3→16→16

卡槽 5 的墨盒变化情况:

0→55→8→8→8→34→34→34→18→44→44→44→28→28→28→28→28→50→50→50→40→40→40→40→40→40→16→16→38→24→24

卡槽 6 的墨盒变化情况:

0→33→10→10→2→12→12→12→12→10→10→10→10→10→10→18→18→18→48→48→48→48→48→12→4→23→17→51→51→42→9

卡槽 7 的墨盒变化情况:

0→14→4→4→4→3→3→51→51→51→51→57→24→24→49→49→41→41→46→46→57→14→14→14→33→30→30→30→6→6→6

卡槽 8 的墨盒变化情况:

0→9→49→35→35→35→48→48→30→26→26→6→6→6→20→36→7→11→10→10→10→10→10→10→46→29→29→59→59→59→39

卡槽 9 的墨盒变化情况:

0→20→19→19→19→24→24→24→20→43→43→50→50→50→30→30→38→8→8→8→51→23→23→49→22→22→48→48→49→13→13

卡槽 10 的墨盒变化情况:

0→26→46→46→46→33→40→40→40→56→56→56→56→56→1→19→58→56→56→56→56→56→56→9→13→13→50→50→10→1→1

附录 5
介绍：问题二的详细求解结果

- **Ins1_5_5_2:**

最优目标值: 56.0

卡槽 1 的墨盒变化情况:
0→3→2→1→3→2
卡槽 2 的墨盒变化情况:
0→2→1→3→1→3

- **Ins2_10_30_10：**

最优目标值: 184.0
卡槽 1 的墨盒变化情况:
0→17→22→22→17→17→24→26→20→7→7
卡槽 2 的墨盒变化情况:
0→29→29→29→29→11→11→5→29→29→29
卡槽 3 的墨盒变化情况:
0→6→6→6→6→6→23→23→26→6→19
卡槽 4 的墨盒变化情况:
0→26→26→17→22→22→22→22→22→23→23
卡槽 5 的墨盒变化情况:
0→23→27→27→3→29→9→9→17→17→17
卡槽 6 的墨盒变化情况:
0→5→15→15→15→15→8→3→18→18→18
卡槽 7 的墨盒变化情况:
0→25→9→11→5→5→5→24→4→25→25
卡槽 8 的墨盒变化情况:
0→8→12→21→21→21→21→15→2→28→28
卡槽 9 的墨盒变化情况:
0→28→23→8→8→26→4→4→13→22→22
卡槽 10 的墨盒变化情况:
0→22→7→7→7→7→7→7→24→24→24

- **Ins3_20_50_10：**

最优目标值: 249.0
卡槽 1 的墨盒变化情况:
0→3→47→1→1→1→8→8→8→8→8→33→6→29→15→15→10→10→42→35→35
卡槽 2 的墨盒变化情况:
0→4→6→33→33→33→33→7→17→17→21→18→18→19→30→47→29→29→29→29→29
卡槽 3 的墨盒变化情况:
0→21→30→30→30→39→39→28→16→16→16→16→16→7→7→7→7→7→45→45→14
卡槽 4 的墨盒变化情况:
0→37→35→35→20→5→5→5→5→46→46→46→46→18→18→18→25→33→33→33→49
卡槽 5 的墨盒变化情况:
0→26→36→36→36→36→43→43→43→43→43→43→43→1→33→33→12→8→34→28→16

卡槽 6 的墨盒变化情况:

0→29→49→49→49→49→4→21→21→21→33→6→21→21→21→35→35→35→10→10→10

卡槽 7 的墨盒变化情况:

0→39→39→39→41→14→14→14→22→34→26→24→49→23→19→19→43→43→43→43→1

卡槽 8 的墨盒变化情况:

0→14→45→17→17→4→6→6→6→38→45→40→40→40→38→25→14→12→8→8→8

卡槽 9 的墨盒变化情况:

0→32→42→42→7→7→7→3→3→3→3→3→9→9→9→4→23→23→23→24→24

卡槽 10 的墨盒变化情况:

0→35→9→26→8→10→10→10→10→10→10→10→28→26→12→34→15→45→41→41→41