第八届华中地区大学生数学建模邀请赛

承　诺　书

我们仔细阅读了第八届华中地区大学生数学建模邀请赛的竞赛细则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的, 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，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们的参赛报名号为：104971172

参赛队员　(签名) ：

队员 1：魏子厚

队员 2：杨劲赫

队员 3：林荣武

武汉工业与应用数学学会

第八届华中地区大学生数学建模邀请赛组委会

第八届华中地区大学生数学建模邀请赛

编 号 专 用 页

选择的题号：A

参赛的编号：104971172

（以下内容参赛队伍不需要填写）

竞赛评阅编号：

第八届华中地区大学生数学建模邀请赛

钢构料的排料问题

摘要

本文就在矩形钢构料排料这道工序中，如何在板料中摆放零件使板料的利用率最高进行了讨论。排料过程中的利用率是指零件面积与一刀切后继续切割的面积的比值。

对于问题一，本文利用一刀切，零件互不重叠，零件不超过边界等约束条件，以及零件尺寸之间的关系，使用 lingo 找出零件的最优组合，将多个零件先拼接为一个，减少零件个数和待拼区域的面积，从而使问题简化 ，在零件个数比较少的情况下，结合贪心算法，建立了整数规划模型，求得目标函数利用率最大值为 97.04%。

对于问题二，本文针对两种不规则零件，对于五边形零件，将其两两对接后再转换为包络矩形，使不规则二维零件排样问题简化并通过减少废料区域的面积增大了利用率；对于凹六边形，将其简化为包络矩形。然后建立了基于粒子群算法的优化模型，针对有限数量的包络矩形零件，在满足一刀切等约束条件下，从众多的排布方式里找到相对最优的排布（切割）方式，计算得最大利用率为 86.68%。

对于问题三，本文首先依据贪心算法的先考虑局部最优解的思想，先对第一块板材实行局部最优化求解，利用 lingo 得到最优线性组合，将多个零件拼接在一起简化为一个零件，得到在局部 100%的利用率下板材一的排列方式，然后利用粒子群优化算法在第二块板材建立了局部最优的整数规划模型，得到最大利用率为 91.45%。

关键词：零件拼接，包络矩形，粒子群算法，贪心算法，Guillotine 切割

1.问题重述

1.1 问题背景

在钢构件制造产品的生产过程中,依照产品零件尺寸从板料中截取大小适当的零件过程称之为排料。排料过程中，不同的排料方案具有不同的材料利用率，而原材料的利用率直接影响产品的成本。因此，降低废料率提高原材料利用率是钢构件生产企业追求的目标。根据实际情况，板材排料又可分为两种：一是规则形状的零件排料，一是不规则形状的零件排料。

规则形状零件是指矩形零件。其描述一般只需用矩形的长和宽。规则形状零

件的排料问题的实质是研究如何组合零件摆放问题，使得在整个原料上摆放大量的不同长和宽的零件产生的废料最少、整料和余料的利用率最高。排放时，其零件间的搭接关系的处理相对容易，只需考虑长、宽两个因素（含预留的损耗量）。

不规则零件在这里是指多边形零件，相对矩形零件排料而言，不规则零件的直接排料要复杂得多。另外由于切割工艺的要求，切割只能实行“一刀切”的工艺（在整料或余料中，从一边的某点到另外一边某点的连线一次切割，但可以在切割下来的板料中再次切割）。板材的利用率就是所有零件面积之和与在一刀切工艺后继续切割的那部分板材面积的比值。

1.2 需要解决的问题

问题 1：对 1 张板料和若干规则形状零件，如何切割可使板料的利用率最高。
问题 2：对 1 张板料和若干不规则形状零件，如何切割可使板料的利用率最高。
问题 3：对 2 张板料和若干规则形状零件，如何切割可使板料的利用率最高。

2．问题分析

问题的关键在于利用率问题，要取得最大利用率，也就是说要在板料上使用“一刀切”的方法，切割出尽可能多的零件,并通过零件的排列使第一次一刀切产生的废料面积最大。问题属于线性规划中的二维下料（板材下料），切割板材时的切割工艺为“一刀切”，即在切割时只能从某边的一点切到另一边的某点。我们将问题转化为在原料上拼接零件，使在满足一刀切的条件下，尽可能多的拼接零件。并满足下列约束条件：

1、零件互不重叠
2、任意零件不能超出材料边界
3、满足 Guillotine 切割的工艺要求

2.1 对问题一的分析

问题一是在一块板材上的二维规则零件的排样最优化问题，我们先利用一刀切的约束条件，以及零件尺寸之间的关系，将多个零件先拼接起来简化为一个，减少零件个数和待拼区域的面积，在零件个数比较少的情况下，就可利用贪心算法求得最优解。

2.2 对问题二的分析

问题二是在一块板材上的二维不规则零件的排样问题，就是在给定的平面区域内找出被排零件的 $P_1,P_2...P_n$ 排放在指定的 P 中，使得材料的利用率最高.

首先把将二维不规则零件简化为二维规则零件,将两个五边形拼接为一个四边形，六边形以矩形近似处理。在零件个数比第一问多 10 个的情况下，不继续利用用第一问的贪心算法，采用粒子群优化算法，寻找每个零件的最优位置，从而实现提高板材利用率的目的。

2.3 对问题三的分析

问题三也是二维规则零件的排布问题，不同于问题一的是，问题三是在两块板料上排布零件，利用贪心算法的思想，我们先在第一块板上实现局部最大利用率，再考虑将剩下的零件排放在第二块板材上，通过恰当的排列方式实现最大的板材利用率。

3.模型假设

（1） 假设不考虑刀片厚度。

（2） 假设不考虑切割过程中板材的损耗。

（3） 假设不考虑板材厚度的影响。

4.符号说明

符号	说明
c	自定义变量，取值为 0,1,2；
A	原料
x	自定义变量，取值为 0,1；
R_j^r	原料 A 上的第 r 个 j 类待排零件
l_j	第 j 类零件的长
w_j	第 j 类零件的宽
Y	A 上零件所有的布局方案
s_f	废料面积
L	原板材的长
n_j	第 j 类零件的个数
W	原板材的宽
k	零件种类数

5. 问题一模型的求解

5.1 零件的预处理简化

单位：毫米

板材：2350*900【1 张】（即 L=900,W=2350）

零件：

表一

	零件 1	零件 2	零件 3	零件 4	零件 5	零件 6	零件 7	零件 8	零件 9
长	350	350	500	500	500	300	250	500	500
宽	300	200	240	210	350	250	200	400	200
个数	2	2	2	2	2	2	2	2	2

我们将问题由在原料上切割零件，转化为在原料上拼接零件，所以，在解决问题之前，先将问题简化，利用多个零件线型组合排列在一起的方式将多个零件简化为一个。解方程：

$$x\ l_i = x\ l_{i'} + x\ l_{i''}$$

$$x\ w_i = x\ w_{i'} + x\ w_{i''}$$

解得：$\begin{cases} l_5 = w_2 + 2l_7 \\ 2w_8 = 2w_1 \end{cases}$

所以将两个 7 号零件与 5 号零件拼在一起简化为 1 个零件，将一个 2 号零件两个 1 号零件和两个 8 号零件拼在一起简化为 1 个零件。拼接后的结果如图：

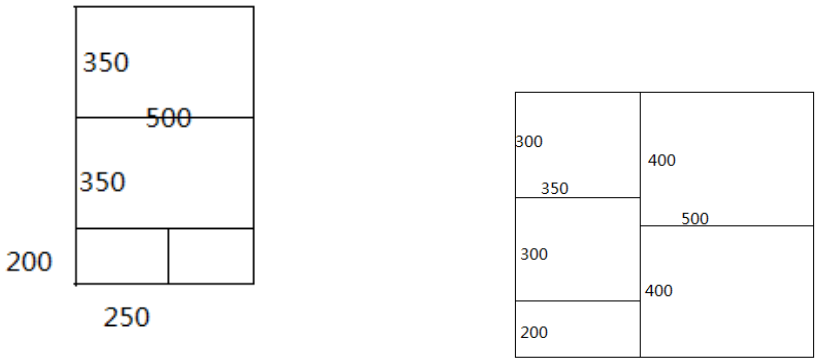
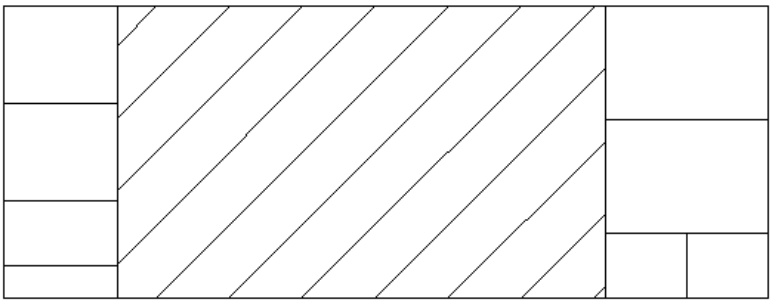


图 1

现在的问题简化为，在阴影区域内拼接如下零件：



表二

	零件 2	零件 3	零件 4	零件 6	零件 9

长	350	500	500	300	500
宽	200	240	210	250	200
个数	1	2	2	2	2

5.2 基于贪心算法的模型最优化求解

（1）贪心算法为，每次选择时，只考虑使当前最优的方案，对于这个问题，我们在拼接零件时首先考虑在零件中选择适当个数，将面积较大的零件拼接进去，使其长度之和最接近原材料的长，则目标函数为：

$$\min l_s - \sum l_i c,$$

$$\min \Delta w$$

经过 lingo 求解，得到的最优组合为 $n_3 = 2 \quad n_4 = 2 \quad n_9 = 2$

所以零件拼接如图：

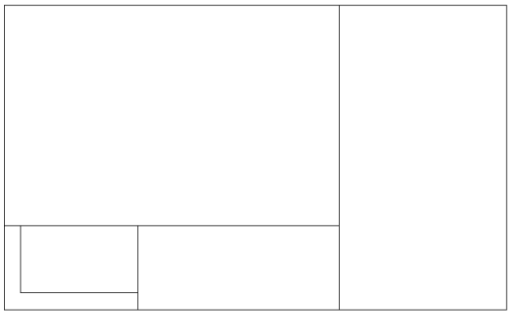
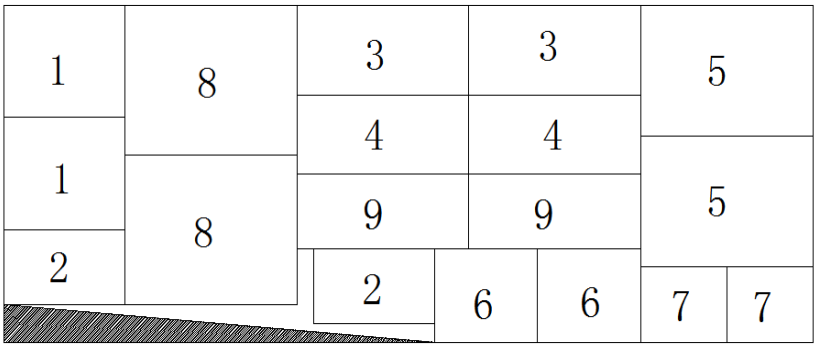


图 2

将剩余部分视为一块新的原材料，长为 $w - w_3 - w_6 - w_9$ ，宽为 w_s，

再将剩余零件拼接，得到最终凭借方式如图：



经计算，利用率为 $f=\dfrac{\sum_{j=1}^{k} n_j l_j w_j}{LW-S_f}=97.04\%$

6.问题二模型建立与求解

6.1 零件的简化预处理

应用 MATLAB，得出两种类型零件的具体形状

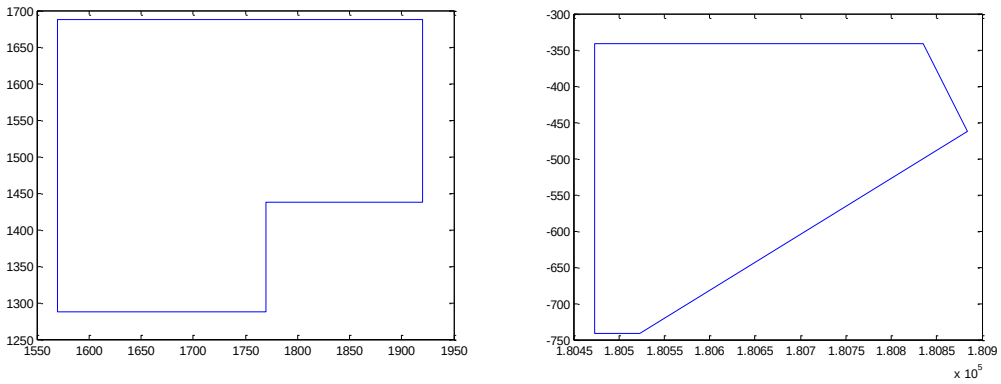


图 4

我们采用求取最小包络矩形的方法进行简化预处理．这样，二维不规则零件排样问题就转化为矩形件排样问题，由于五边形的种类只有一种，所以我们将两个五边形沿斜边拼接在一起再包络，以任意一条边向两边延伸，作为包络矩形的一条边，以不规则零件中相对于这条边的最外面的两个顶点(或者曲线段的外切点)，向这条边做垂线并向上延伸，以离这条边最远的顶点(或者曲线段的外切点)为端点，做这条边的平行线，这样就构成了一个包络矩形．每条边依次如此构造

一个包络矩形，其中面积最小的一个就认为是最小包络矩形。如图 5：

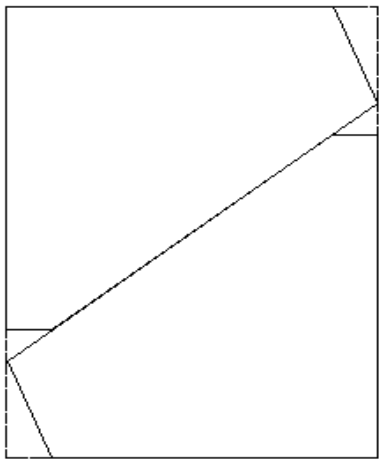


图 5

凹六边形近似为矩形处理，如图 6:

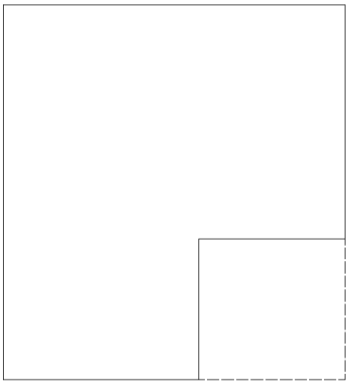


图 6

6.2 模型的建立与约束条件

最优化模型为 $\max f = \dfrac{\sum_{j=1}^{k} n_j l_j w_j}{LW - S_f}$ （L 为原板材的长，W 则为宽）

利用平面点集，A 代表原料， $A = \{(\bar{x}, \bar{y}) \mid 0 \le \bar{x} \le L, 0 \le \bar{y} \le W\}$

R_j^r 表示：原料 A 上的第 r 个 j 类待排零件，设其左下角坐标（x,y），λ 表示横排或竖排（横为 1，竖为 0）。

则 $R_j^r = R_j^r(x, y, \lambda) = \{x \le \bar{x} \le x + (1-\lambda)w_j + \lambda l_j, y \le \bar{y} \le y + (1-\lambda)l_j + w_j\}$

约束条件：1. 零件不能有超过原材料的部分；

2. 任意两个零件不能重叠；

3. 满足“一刀切”；

约束条件 1、2 的数学表达式如下 ：

$$\begin{cases} R_j^r \in \mathrm{A}, & 1 \le r \le n_j \\ R_j^r \cap R_j^{r'} = \varnothing & r' \ne r \end{cases}$$

约束条件 3 的数学表达式如下：

Y 表示 A 上所有的布局方案，Y' 为子方案，P（Y'）=$\left\{Y' \mid Y' \subseteq Y, Y \ne \varnothing\right\}$ ，$I_n = \{0,1,2.....n\}$。对任意$Y' \in \mathrm{P(Y)}$，当$\left|Y'\right| \ge 2$，存在正实数t_0，坐标向量$e_\eta, k \in I_2$，使得$Y' = Y'_- \cup Y'_+$，Y'_-，$Y'_+ \in \mathrm{P(Y)}$，满足

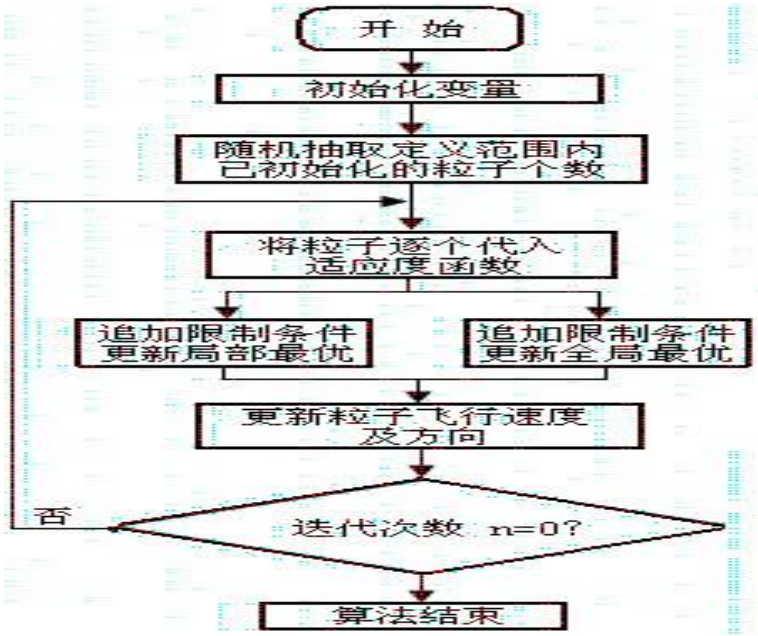
$$\begin{cases} \max\left\{\langle \eta, e_h \rangle \middle| \eta \in R_{ij}^r, \forall R_{ij}^r \in Y'_- \in P(Y)\right\} \le t_0 \\ \max\left\{\langle \eta, e_h \rangle \middle| \eta \in R_{ij'}^{r'}, \forall R_{ij'}^{r'} \in Y'_+ \in P(Y)\right\} \ge t_0 \end{cases}$$

其中，$e_h = e_1$时，$t_0 < \mathrm{L}$，$e_h = e_2$时，$t_0 < \mathrm{W}$，若任意$Y' \in \mathrm{P(Y)}, \left|Y'\right| \ge 3$，$\exists t_0 > 0$，$e_h$ 使两式同时成立，则 Y 为 A 上的一刀切方案。

综上所述，整理出来的约束条件如式：

$$\text{S. t.} \begin{cases} \sum_{i=1}^{m} n_{ij} = n_j, \quad j \in I_n \\ \bigcup_{j=1}^{n_j} \bigcup_{r=1}^{n_{ij}} R_{ij}^r \subseteq A_i, \quad i \in I_n \\ \mathrm{int}\, R_{ij}^r \cap \mathrm{int}\, R_{ij}^{r'} = \varnothing, \ R_{ij}^r, \ R_{ij'}^{r'} \in Y, \ (r, j) \ne (r', j') \\ \max\left\{\langle \eta, e_h \rangle \middle| \eta \in R_{ij}^r, \forall R_{ij}^r \in Y'_- \in P(Y)\right\} \le t_0 \\ \max\left\{\langle \eta, e_h \rangle \middle| \eta \in R_{ij'}^{r'}, \forall R_{ij'}^{r'} \in Y'_+ \in P(Y)\right\} \ge t_0 \\ Y'_- \cup Y'_+ = Y' \in P(Y) \end{cases}$$

6.3 基于粒子群算法的模型最优化求解



粒子群算法

POS 算法中，每个优化问题的解就是解空间中一个粒子，而在本体中布局方案就是粒子，所有的粒子都追随当前最优粒子，通过迭代在解空间中找到最优解。在每一次迭代中，粒子通过跟踪两个极值更新自己，一个是自己本身找到的最优解，这个解称个体极值 P_{best}；另一个极值是整个解空间的最优解 q_{best}，这个解称为全局最优解，全局模式 POS 算法与本题分为六个步骤：

步骤 1：采用随机产生的位置和速度在整个解空间中初始化粒子群。初始化中，粒子代表解空间的一个解，粒子的长度由零件数量确定，排样零件的不同顺序组合构成不同的粒子，调换其中一些零件的顺序即变为另一个粒子，赋予每一个粒子一个随机速度作为初始速度。粒子 P=[P1.....Pn],其中，Pi 表示第 i 个零件的左下角和右上角的坐标。

步骤 2：结合本文中的目标函数与约束条件，算出粒子的适应度函数。对于每个粒子，根据适应度函数计算其适应度函数值在计算适应度函数值之前，先根据约

束条件判断粒子位置是否满足条件，如果不满足，对粒子位置微调。

步骤 3：比较每个粒子的适应度函数值和 P_{best}。如果当前值优于 P_{best}，就设置当前值为新的 P_{best}，粒子当前位置为新的 P_{best} 的位置 x_{id}。比较适应度函数值时，需根据具体的优化问题决定。可能数值越大越优，也可能恰恰相反。

步骤 4：比较每个粒子的适应度函数值和 q_{best}，如果当前值优于 q_{best}，就设置当前值为 q_{best}，粒子的当前位置 x_{id} 设置为 q_{best} 的位置 X_{id}^{g}。

步骤 5:按照式（1）和式（2）改变每个粒子的速度和位置：

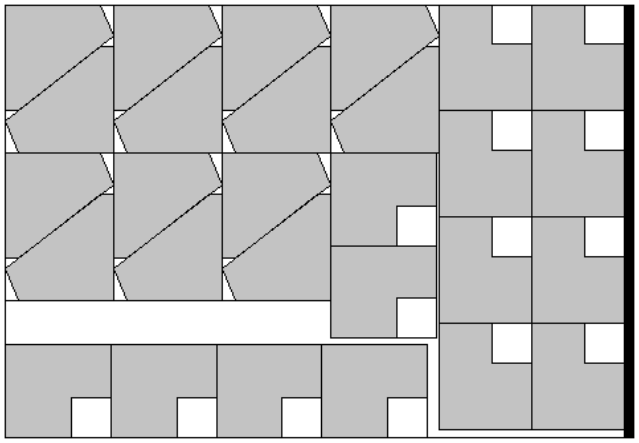
$$\begin{aligned} v_{id} &= K \cdot [v_{id} + c_1 \cdot \mathrm{rand}_1() \cdot (X_{id}^{p} - x_{id}) + \\ &\quad c_2 \cdot \mathrm{rand}_2() \cdot (X_{id}^{g} - x_{id})] \qquad (1) \\ x_{id} &= x_{id} + v_{id} \qquad (2) \end{aligned}$$

其中，c_1 和 c_2 为学习因子，rand1（），rand2（）位[0,1]区间内的随机数；M_{id} 和 x_{id} 分别为粒子当前的速度和位置；粒子的速度 M_{id} 受到最大速度 $V_{\max}$ 的限制，如果超过了最大速度，就限定为最大速度。K 为压缩因子，本体计算数据不大，故没有采用压缩因子方法的 PSO 算法。

$$K = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \qquad (3)$$

在采用压缩因子的 PSO 算法中，一般取 $c_1 = c_2$ =2105,K=01729，此外，取最大速度为粒子的动态范围可显著提高 PSO 的性能。

步骤 6：如果满足停止循环的条件，就终止计算，得出最优解；否则转到步骤 2 继续循环。停止循环的条件为满足迭代次数，对于本题，我们定迭代次数为 300。切割效果如图，黑色阴影区为废料：



则 $f=\dfrac{\sum_{j=1}^{k} n_j l_j w_j}{LW-S_f}=86.68\%.$

7.问题三模型建立与求解

7.1 零件与处理

首先，我们仍按照问题一的处理方法简化问题：

使方程 $x\,l_i = x\,l_{i'} + x\,l_{i''}$ 成立，

并使 $(w_s - \sum w_i c_i)\min$

解得：$\begin{cases} l_5 = l_4 = 2l_6 \\ l_5 = l_8 = l_3 \end{cases}$

所以，我们按照线性组合的方式将多个零件简化为一个，如图 8：

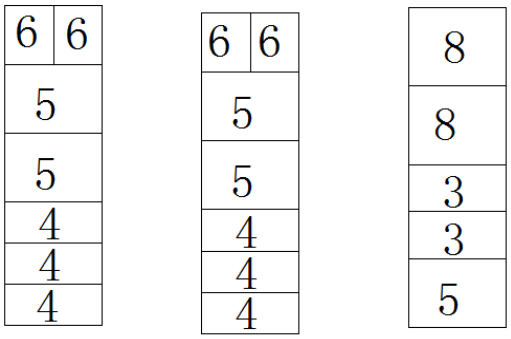


图 8

将简化后的零件拼入第一块板，可使当前板一的利用率最最大。拼接效果如图：

图 9

此刻，已经使第一块板的板材利用率最大，然后考虑将剩余零件拼入第二块板，使第二块板的板材利用率最大。

7.2 基于 BL 策略和粒子群算法的最优化求解

BL 策略的定义为：在排样过程中，保证样本之间两两不重叠，一个排样单元从右上方向左下方移动，当不能向下或向左移动一步时，那么，他就达到了它的一个 BL“稳定位置”。

约束条件与问题二相同：

$$
\text{S. t.}\quad
\begin{cases}
\sum_{i=1}^{m} n_{ij} = n_j, \quad j \in I_n \\
\bigcup_{j=1}^{n_j} \bigcup_{r=1}^{n_{ij}} R_{ij}^{r} \subseteq A_i, \quad i \in I_n \\
\text{int } R_{ij}^{r} \cap \text{int } R_{ij'}^{r'} = \varnothing, \ R_{ij}^{r}, \ R_{ij'}^{r'} \in Y, \ (r, j) \neq (r', j') \\
\max\left\{ \langle \eta, e_h \rangle \middle| \eta \in R_{ij}^{r}, \forall R_{ij}^{r} \in Y_{-}' \in P(Y) \right\} \leq t_0 \\
\max\left\{ \langle \eta, e_h \rangle \middle| \eta \in R_{ij'}^{r'}, \forall R_{ij'}^{r'} \in Y_{+}' \in P(Y) \right\} \geq t_0 \\
Y_{-}' \cup Y_{+}' = Y' \in P(Y)
\end{cases}
$$

第三个问题零件数量仍旧比较多，所以考虑用粒子群优化算法求取最优位置，算法的相应步骤如问题二，在这里利用率为 $f = \dfrac{\sum_{j=1}^{k} n_j l_j w_j}{LW - S_f}$，废料面积的值受到一刀切的切割方向影响，为使问题简化，我们不考虑斜切，只考虑横切与竖切。

最终原料板二的切割方案为：

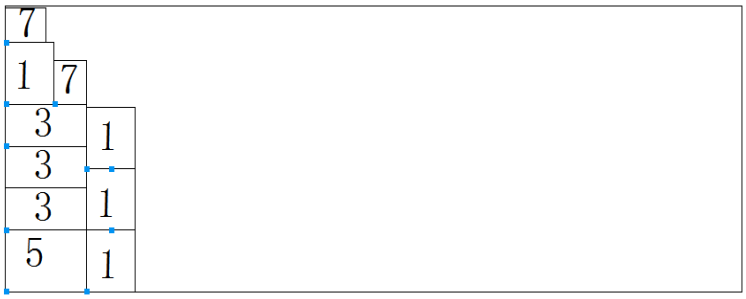


图 10

综上，问题三的切割方案为：

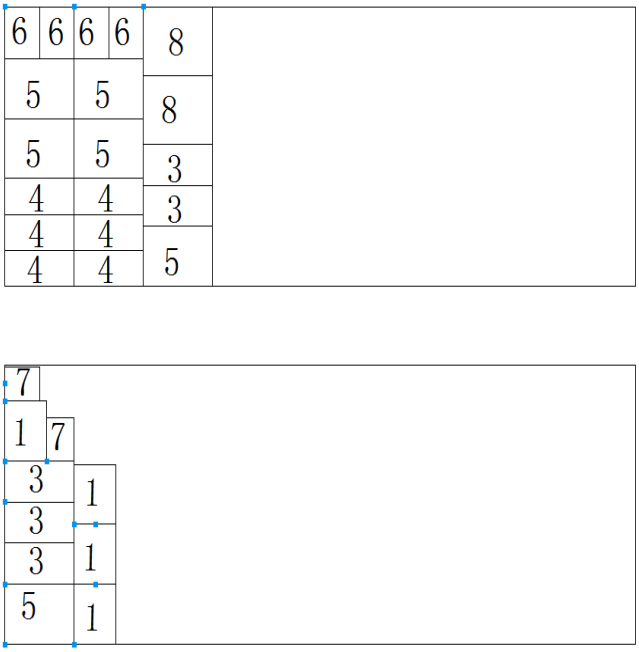


图 11

利用率为 $f=\dfrac{\sum_{j=1}^{k} n_j l_j w_j}{LW-S_f}=91.45\%$

8.模型结果的检验

对于问题一、三，使用了专门的下料软件进行评估，模型所得结果与软件分析优化结果误差很小；对于问题二，使用 AutoCAD 和 ProNest，精确地验证了结果。

9.模型评价

1.模型优点：

（1）.我们将排料问题转化为零件的拼接问题，所需考虑的利用率最大的问题转化为在满足一刀切约束下将零件尽可能多的拼接在原料上，使第一次一刀切的废料最多。

（2）.拼接零件时，将多个不同的零件拼接在一起简化为一个零件，减少了零件的个数，从而降低了运算复杂度。

（3）充分利用了 MATLAB、lingo，采用了贪心算法和粒子群算法，得到的数据较准确。

2.模型缺点:

（1)模型忽略了现实操作中的影响因素，如光，风，湿度等，以及切割时不可避免的损耗所带来的误差；

（2）粒子群优化算法具有易陷入局部极值，过早收检，收检性差等特点，导致得出的解与题目真正的最优解存在误差；

10.参考文献

【1】姜启源、谢金星,《数学建模》，北京，高等教育出版社，2011.8；

【2】薛定宇、陈阳泉，《高等应用数学问题的 MATLAB 求解》，北京，清华大学出版社，2013.1；

【3】司守奎、孙玺菁，《数学建模算法与应用》，北京，国防工业出版社，2013.2

【4】 李明，张光新，周泽魁， 基于改进遗传算法的二维不规则零件优化排样 湖南大学学报 第 33 卷第 2 期 2 0 0 6 年 4 月

【5】李蒙 ，基于一刀切的多原材二维下料协同优化方法研究 河北工业大学 2012.11

【6】石晨，板材优化下料算法的研究 同济大学 2007.3

【7】李明，宋成芳，周泽魁， 二维不规则零件排样问题的粒子群算法求解 江南大学学报 第 4 卷第 3 期 2005 年 6 月

附录（含代码）

1.粒子群算法代码

```
%% 产生初始粒子和速度
for i=1:sizepop
    %随机产生一个种群
    pop(i,:)=2*rands(1,2);      %初始种群
    V(i,:)=0.5*rands(1,2);   %初始化速度
    %计算适应度
    fitness(i)=fun(pop(i,:));     %染色体的适应度
end
%% 个体极值和群体极值
[bestfitness bestindex]=max(fitness);
zbest=pop(bestindex,:);     %全局最佳
gbest=pop;      %个体最佳
fitnessgbest=fitness;     %个体最佳适应度值
fitnesszbest=bestfitness;     %全局最佳适应度值
%% 迭代寻优
for i=1:maxgen

    w=we*(ws/we)^(1+k/maxgen);
```

```
for j=1:sizepop

    %速度更新
    V(j,:) = V(j,:) + c1*rand*(gbest(j,:) - pop(j,:)) + c2*rand*(zbest - pop(j,:));
    V(j,find(V(j,:)>Vmax))=Vmax;
    V(j,find(V(j,:)<Vmin))=Vmin;

    %种群更新
    pop(j,:)=pop(j,:)+V(j,:);
    pop(j,find(pop(j,:)>popmax))=popmax;
    pop(j,find(pop(j,:)<popmin))=popmin;

    %适应度值
    fitness(j)=fun(pop(j,:));

end

for j=1:sizepop

    %个体最优更新
    if fitness(j) > fitnessgbest(j)
        gbest(j,:) = pop(j,:);
```

```
            fitnessgbest(j) = fitness(j);
        end

        %群体最优更新
        if fitness(j) > fitnesszbest
            zbest = pop(j,:);
            fitnesszbest = fitness(j);
        end
    end
    yy(i)=fitnesszbest;

end
%% 结果分析
for m=1:300
    s(101,m)=sum( s(:,m) )/100;
end
plot(s(101,:),'-y')
title('最优个体适应度','fontsize',12);
xlabel('进化代数','fontsize',12);ylabel('适应度','fontsize',12);
```

2.适应度函数 function result=fun(x)

```
L=2380
```

```
W=1630
w=[399,399,399,399,399,399,399,399,399,399,399,399,399,399,350,350,350
,350,350,350,350,350,350,350,350,350,350,350,350,350,350,350,350,]
l=[411,411,411,411,411,411,411,411,411,411,411,411,411,411,400,400,400
,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,]
if (symsum(w(i)*x(i),i,1,28)<=W)&(symsum(l(i)*x(i),i,1,28)<=L)
b=l.*x'
h=max(b)
result=max(symsum(W(i)*l(i)*x(i),i,1,28)/Wh)
end
```