基于 0-1 规划和贪心算法的电子线路板布线优化模型

摘要

本文从规划角度出发，研究了在一定限制条件下构成单个电子元件所需的最少电子线路板个数，并给出了相应限制下的布线建议。由于可行的排线方式复杂多样，难以穷举，我们首先通过分析手段对排线方式进行了筛选，最后建立了一套以 0-1 规划为核心，贪心算法为求解工具的电子线路板布线优化模型，并进行了进一步分析。

针对任务 1，该情况只需考虑 $12cm \times 12cm$ 规格线路板各端口间连接路径为直线的不相交布线模型，本文以每个线路板中各线路的连接情况为决策变量，以线路板为最少目标函数，以在同一线路板内线路的相交限制为约束条件，根据 0-1 规划的思想建立单目标规划模型。在分析各连接线路的相交情况后，列出各线路间的相交情况，并以此为基础建立以贪心算法为核心的启发式算法求得近似最优解为 3 块线路板，并给出布线建议和结果分析。

针对任务 2，该情况需在任务 1 的基础上考虑直线路径可相的交布线模型。首先，在任务 1 的基础上，更改约束条件“线路板上连接线路均不相交”为“线路板上任意两条线路之间可至多存有一个交点”，之后同样以每个线路板中各线路的连接情况为决策变量，线路板为最少目标函数，建立单目标规划模型。在分析连接线路相交情况后，列出各线路间的相交情况，并建立合适的启发式算法寻得最少线路板个数为 2，给出布线建议和结果分析。

针对任务 3，该情况需考虑曲线路径规划的转弯布线模型。先从确定两连接端口之间的优化路径入手，提出了四种相对位置下端口之间相连的较优策略。之后将所有需要连接的端口都以最优方式连接，确定各线路的叠加线路图和各线路间的相交情况，再以每个线路板中各线路的连接情况为决策变量，线路板为最少目标函数，更改约束条件为“线路板内任意一条线路上所存交叉点个数和转弯次数之和不超过 2”，建立单目标规划模型，通过贪心算法得到近似最优解为 3，并给出布线建议和结果分析。

针对任务 4，需要更改任务 1 至 3 中线路板尺寸为 $18cm \times 18cm$，应用先前所建立的模型对三种情况作再次分析。得到三种情况对应的最少线路板个数分别为 6,4,6 个，并对每种情况都给出了相应的布线建议。

最后，通过对建模过程和结果的分析，探讨了本文中所建立模型的优缺点，并为模型的进一步改进优化提供了建议。

1 问题的提出和重述

1.1 问题的提出

在大多数电子设备中，为实现电路中各个元件之间的电气互连，都要使用电子线路板。一个好的布线方案不仅能缩小整机体积，也可以降低成本，提高电子设备的质量和可靠性。所以研究电子线路板中的布线问题就有着重要的意义。

1.2 问题的重述

在一个 $12cm \times 12cm$ 的矩形电子线路板上，有 $A1 \sim A5, B1 \sim B5, C1 \sim C5, D1 \sim D5$ 共计 20 个端口。现有某种电子零部件需要将这些端口按表 1 相连。若连接时导线无法按照要求布置上。则可放置几个完全相同的电子线路板并让线路板的同名端口用导线贯穿连接。若端口在某一层有按连接表所规定的方式相连，则认为这两个端口在此电子零部件中连接在了一起。

起始端口	A1	B2	A2	D1	B4	A4	C2	D3	C4	C5
终止端口	B1	C1	A5	B3	A3	B5	D2	C3	D5	D4

Table 1

任务 1：端口可直线相连，但导线彼此不能相交。建立模型并求解最少需要的电路板个数。
任务 2：端口可直线相连且任意导线至多与其他导线相交一次。求最少需要的电路板个数。
任务 3：端口相连时，导线需垂直于两起始端口。其转弯半径不能小于 1cm，转弯角度需是 90° 的整数倍并且导线的交叉数和转弯数不大于 2. 求解最少需要的电路板个数。
任务 4：利用先前建立的模型重新讨论 $18cm \times 18cm$ 矩形线路板上的对应排线方案。

2 问题的分析

针对任务 1，两个特定端口是否有导线连接，可用 0-1 逻辑变量表示，以所需电子线路板的个数为目标函数，以每块板上的连接情况设立 0-1 决策变量，以各任务中端口的连接次数和导线相交情况作为约束条件，建立以 0-1 单目标规划为基础的电子线路板布线模型。并且根据规划思想，给出具有可行性的优化算法。

针对任务 2，在任务 1 所设前提下，增加了在同一块线路板上，每条导线可至多相交一次的条件。可以任务 1 中建立的模型为基础，进一步弱化约束条件，之后同样以 0-1 规划思想建立单目标规划模型，得到在新条件下设立的电子线路板布线模型，同时给出相应的优化算法。

针对任务 3，我们首先根据输入和输出端口的相对位置，讨论了当两个相连端口分别在线路板的对边，邻边和同边上的连接情况，并设立了总计 5 种以转弯次数最少为基础的优化路径。并以此为基础，讨论不同路径的相容程度。同时，继承任务 1 和 2 的分析模式，建立规划模型，给出具有可行性的算法。

针对任务 4，将任务 1 至 3 中所建立的模型应用于更复杂的 32 端口布线情形，同时对原模型进行进一步评价与讨论。

3 模型的假设

3.1 模型的假设

1、在可以直线连接两端口的前提下，应尽量选择直线连接对应端口。

2、线路板可看作一张没有额外厚度的平面且其上连接对应端口的导线均可以看作平滑的连续曲线，导线宽度忽略不计。

3、所有电子线路板大小和规格完全一致，且按照上下端口同名的状态垂直堆叠放置。

4、每种对应的输入输出端都相连有且仅有一次。

4 符号及变量说明

符号	表示含义	单位
N	最终所需最少电子线路板的个数	个
N_t	电子线路板的个数	个
i	$1 \sim N_t$ 个板的编号	
j	$1 \sim 10$ 种连接方式	
x_{ij}	i 号板 j 方式的连接情况	
ρ_i	第 i 块板上的利用率	
r_{mn}	在同一块电子板上第 m 种连接方式和第 n 种连接方式相交次数	
R	排斥矩阵: 第 m 行 n 列为 r_{mn}	
t_m	第 m 种排线方式的转弯次数	

5 模型的建立

5.1 不相交布线模型的建立与求解

5.1.1 模型的建立

任务 1 中，端点可直接相连且导线不相交。首先给每一种连接方式编号 $1 \sim 10$(见表 2)，并且由于第 i 块板上的第 j 种连接方式只有有与没有两种情况，所以引入 0-1 **决策变量** x_{ij}, 则有

$$x_{ij} = \begin{cases} 0, \text{i 板中没有第 j 种连接方式} \\ 1, \text{i 板中有第 j 种连接方式} \end{cases} \tag{1}$$

线路编号	1	2	3	4	5	6	7	8	9	10
起始端口	A1	B2	A2	D1	B4	A4	C2	D3	C4	C5
终止端口	B1	C1	A5	B3	A3	B5	D2	C3	D5	D4

Table 2

目标函数为：

$$min\ N$$

约束条件为：

(1) 在第 N_t 块板之前，第 j 种连接方式至多出现一次

$$\mathrm{x}_j^{(N_t)} = \sum_{i=1}^{N_t} x_{ij} \leq 1$$

(2) 在任意一块板上，都不可以有两条导线相交

$$\mathrm{r}_i = \sum_{m=1}^{10} \sum_{n=1}^{10} x_{im} x_{in} r_{mn} = 0$$

(3) 在最终所求电子线路板之前，所有连线方式出现且仅出现一次。

$$x_j{}^{(N)} = 1$$

综上，所需最少电子线路板数量的模型为：

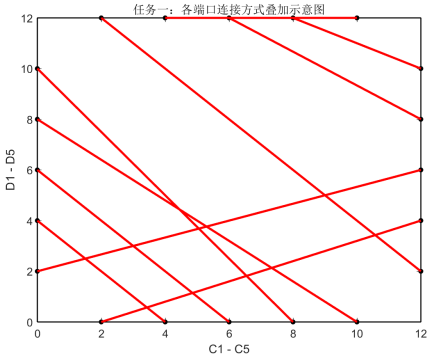
$$min\ N$$

$$s.t. \begin{cases} x_j{}^{(N_t)} = \sum_{i=1}^{N_t} x_{ij} \leq 1 \\ r_i = \sum_{m=1}^{10} \sum_{n=1}^{10} x_{im} x_{in} r_{mn} = 0 \\ x_j{}^{(N)} = 1 \end{cases} \tag{2}$$

为了衡量每块线路板的利用程度，我们定义 $\rho_i = \frac{\sum_j^n x_{ij}}{10}$ 为每块线路板的利用效率。

5.1.2 模型的求解算法

首先，根据所需连接的线路和端口之间可以直线相连的假设，我们可得到如图所示的叠加线路图与相应的排斥矩阵，该矩阵的第 m 行 n 列元素表示 m 和 n 号连接线路之间交点的个数。由于可以直线相交，两条线路之间的交点个数最大为 1。



0	1	0	1	0	0	0	0	0	0
1	0	0	0	0	0	1	1	1	1
0	0	0	0	1	1	0	0	0	0
1	0	0	0	0	0	1	1	1	1
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1
0	1	0	1	0	0	0	0	1	0

Figure 1: 任务 1 叠加线路图及对应排斥矩阵

当端口数量较大时，难以逐个列出所有可能出现的线路排布情况，考虑到算法的普适性，在这个模型中，我们尝试使用贪心算法来求得局部最优解。

现从利用率为 0 的时刻开始，预设初始线路板个数为 1，在每次循环中，以当前所有可选择的排线方式里可能产生交点个数最少为依据选取新增线路，并将其对应的决策变量从 0 更改为 1。当剩余选择中不存在可与当前线路板上所有线路相容的选项时 (及所有可选线路都会与已存线路相交)，考虑增添新的线路板。反复执行上述操作，当所有端点都被连上且导线之间互不相交时，终止循环。求解流程图如下。

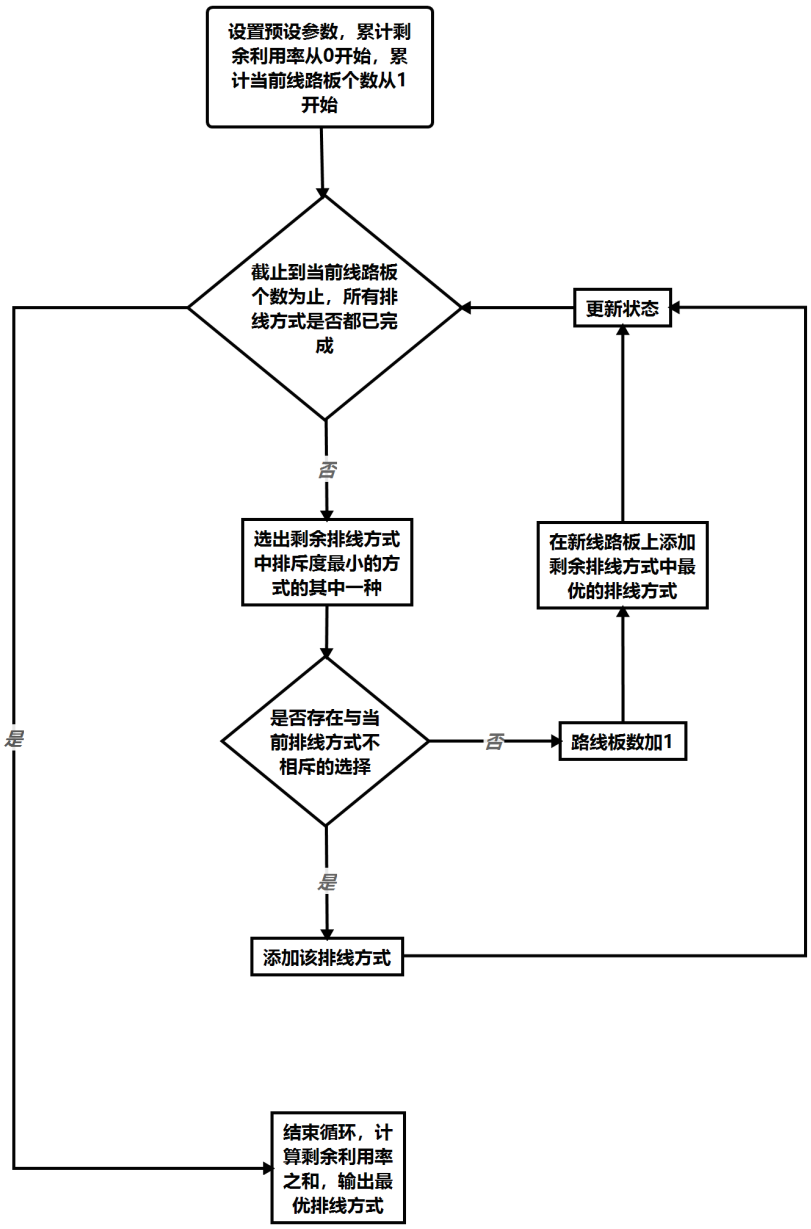


Figure 2: 流程图

使用 matlab 对模型算法进行求解 (具体代码见附录)，可得到以下结果：

最少使用三块线路板 (min N = 3)。第一块板上选择 1，5，6，7，8，9 号连接方式。第二块板上选择 3，10 号连接方式。第三块板上选择 2，4 号连接方式。各个线路板的利用率

分别为 0.6,0.2,0.2，具体连接方式见下图。

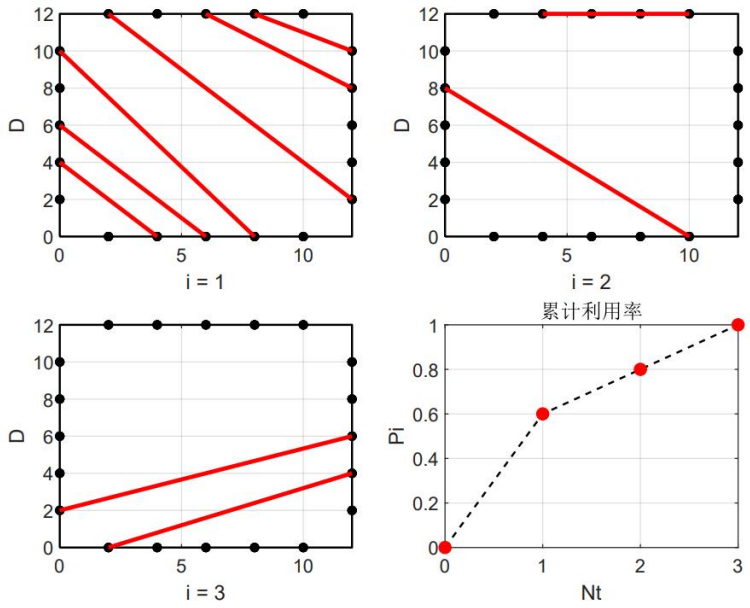


Figure 3: 任务 1 结果

5.2 可相交布线模型的建立与求解

5.2.1 模型的建立

任务 2 中，端点可直接相连且导线至多有一个交点，有: 保持模型 1 中目标函数与其他约束条件不变，条件 (2) 改为在任意一块板上，两条导线至多相交一次:

$$r_i m = \sum_{n=1}^{10} x_{im} x_{in} r_{mn} \leq 1,\ m = 1 \sim 10$$

综上，所需最少电子线路板数量的模型为:

$$min\ N$$

$$s.t. \begin{cases} {x_j}^{(N_t)} = \sum_{i=1}^{N_t} x_{ij} \leq 1 \\ r_i m = \sum_{n=1}^{10} x_{im} x_{in} r_{mn} \leq 1,\ m = 1 \sim 10 \\ {x_j}^{(N)} = 1 \end{cases} \tag{3}$$

5.2.2 模型的求解算法

和不相交的模型类似，从利用率为 0 的时刻开始，综合考虑所有端点是否连上以及导线是否相交来判断是否添加新的电子线路板。反复执行上述操作，当所有端点都被连上且导线之间至多相交一次时，终止循环。流程图如下:

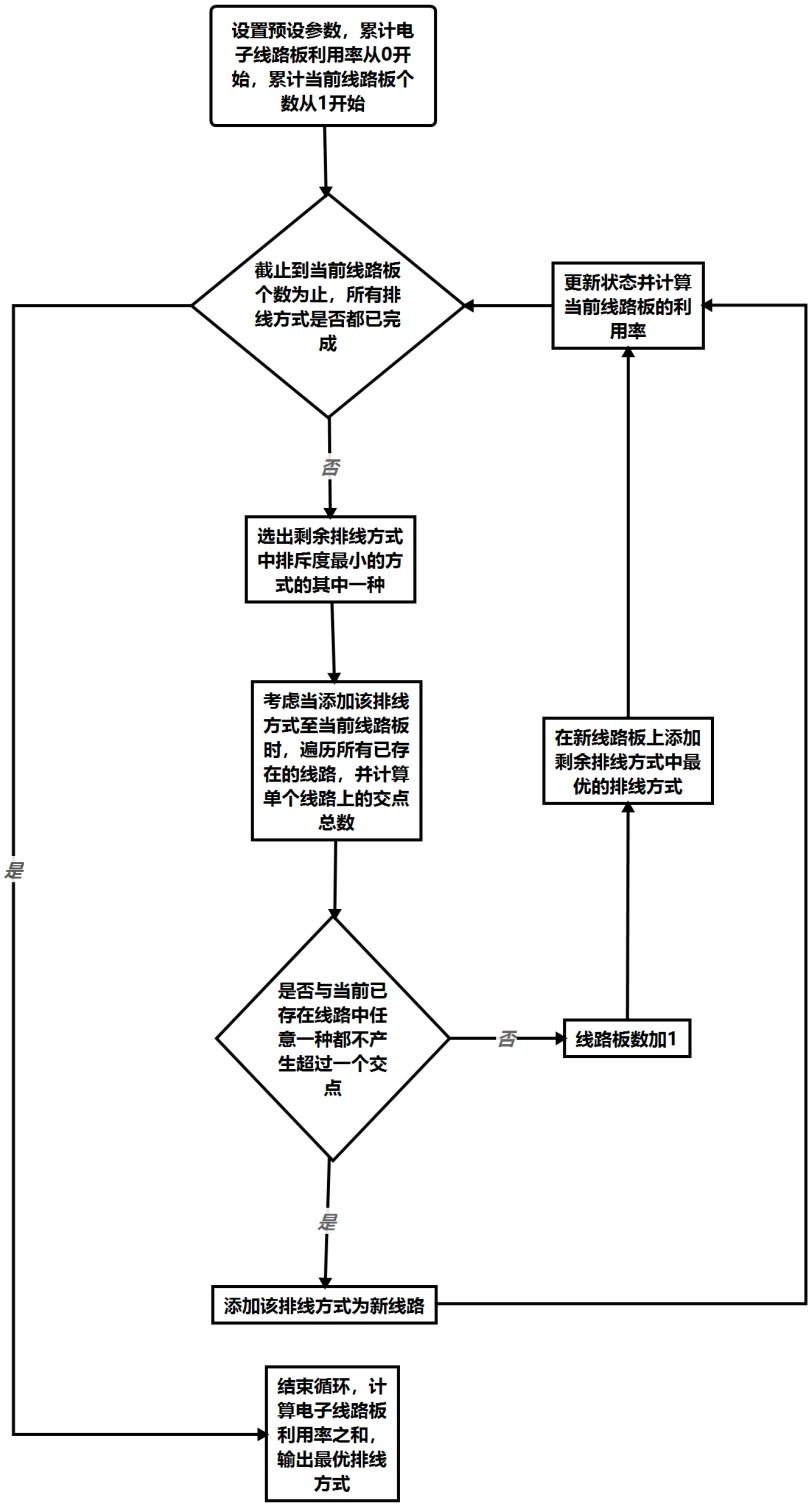


Figure 4: 流程图 2

使用 matlab 进行求解 (具体代码见附录), 解得 $N = 2$。第一块板上选择 1, 5, 6, 7, 8, 9, 10 号线路进行连接。第二块板上选择 2, 3, 4 号线路连接。各板利用率分别为 0.7,0.3。

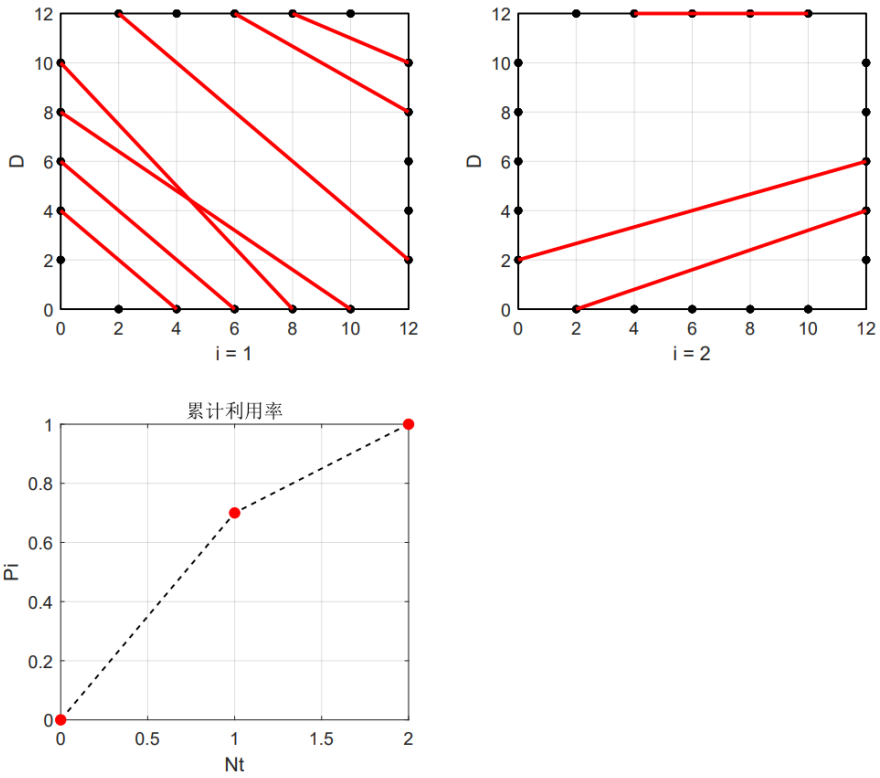


Figure 5: 任务 2 结果

5.3 转弯布线模型的建立与求解

在任务 3 中，端点之间不再是以直线相连，而是以一根垂直于端点所在边界的导线相连。导线可以拐 90° 整数倍的弯，且交叉点和拐弯点之和不大于 2。由于导线的相交情况不唯一且难确定，我们从讨论路线的拐弯次数入手，并假设始末端相同的线路中拐弯次数更少的线路较优，因为这一方案确保了同一根导线上能够容纳更多的交叉点，从而实现当前线路板的更高利用率。在确定了每两个需要连接的端口之间的最优连接方式之后，将问题化为与前文类似的问题进行规划和编程求解。

5.3.1 新增假设

1、始末点相同，拐弯次数少的线路更优。

2、始末点相同，拐弯半径越大的线路越优。

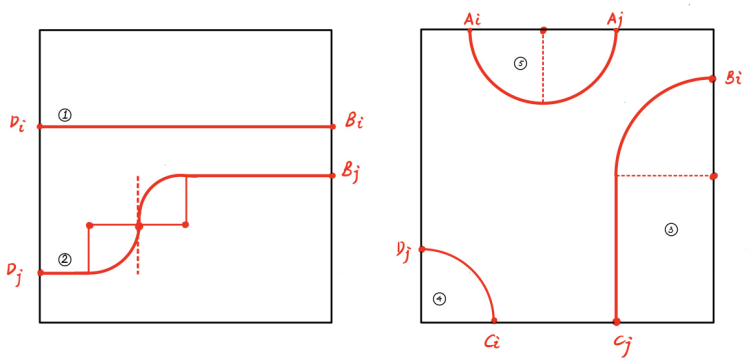
3、一条线路上若存在和其他多条线路的公共点，则该点仅视为一个交叉点。

5.3.2 基于相对位置的优化路径选取策略

以单个线路板的左下角为原点 O，与其相接的两边分别为 x 轴和 y 轴建立平面直角坐标系。由于端口只能排布在矩形线路板的边缘，在同一个平面内，两端口之间的相对位置有且仅有以下四种：

平行 ①	处于对边且连线与线路板边缘平行
对边 ②	处于对边但与线路板边缘不平行
邻边 ③	两端口各自所处的边在线路板上相邻
同边 ⑤	两端口处于线路板的同一边

Table 3



注: 情况 4 为情况 3 中的一种特殊情况

Figure 6

对于“对边平行”的情况，使用直线相连即为符合条件的最优路线，此时，路线上允许存在的交叉点个数最多，为 2。对于“对边不平行”的状况，考虑到问题的复杂性，放在本节末尾进行讨论。下从“邻边”的情况开始讨论：

1) 对于输入输出端口在邻边的情况，基于较优路径拐弯较少的假设，我们选择只拐一次 90° 来让导线可以垂直连入 (连出) 的同时，也能获得更多的交点容纳空间。这里我们以在 AB 邻边上的两个端口为例，假设输入端口和输出端口的坐标分别为 $(a_1, a_2), (b_1, b_2)$, 选取以 o_{ab} 为圆心，$min\{|a_1 - b_1|, |a_2 - b_2|\}$ 为半径 r 的 1/4 圆弧及一条连接圆弧和端点的直线来作为我们的优化路径。接下来求解两种可能情况的解析式：

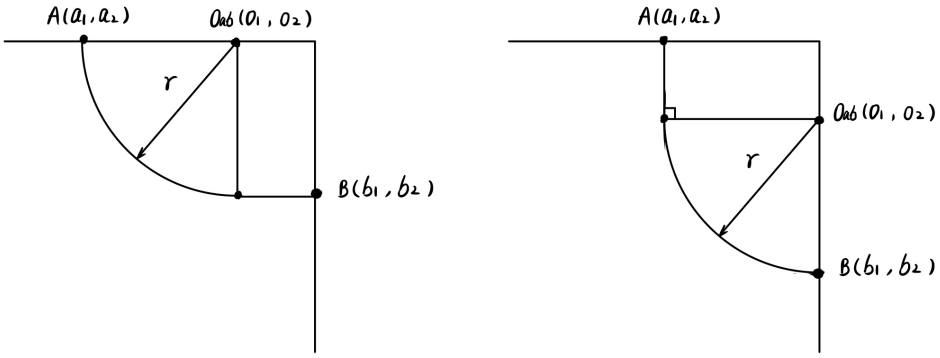


Figure 7

(I) $r = |b_2 - a_2|$, 则圆心坐标为 O_{ab} $(a_1 + r, a_2) = (o_1, o_2)$, 则可得到曲线方程

$$\begin{cases} y = -\sqrt{r^2 - (o_1 - x)^2} + o_2 & , \ a_1 \leq x \leq o_1 \\ y = b_2 & , \ o_1 < x \leq b_1 \end{cases} \tag{4}$$

(II) $r = |b_1 - a_1|$, 则圆心坐标为 O_{ab} $(b_1, b_2 - r) = (o_1, o_2)$, 则可得到曲线方程

$$\begin{cases} x = a_1 & , \ \mathrm{o_2{<}y} \leq a_2 \\ x = o_1 - \sqrt{r^2 - (o_2 - y)^2} & , \ b_2 \leq y \leq o_2 \end{cases} \tag{5}$$

同理可得 BC,CD,AD 邻边的公式 (见附录)。

2) 对于输入输出端口在同边的情况，导线可以通过拐一次 $2 \times 90°$ 来到达另一个端口。此时，我们取 $O_k, k \in \{a, b, c, d\}$ 为圆心，r 为半径的一个半圆来作为优化路径，注意到两个同边位置端口之间的距离最小为 $2cm$, 所以拐弯半径必大于 $1cm$。接下来解出所有可能出现的四种情况的解析式。下列表达式中，用 w 表示线路板的宽度，l 表示长度, 在这里，$l = w = 12$。

(I) 在 A 号边上: 圆心坐标为 $O_a(\frac{a_{i1}+a_{j1}}{2}, w)$, 曲线方程为 $y = o_2 - \sqrt{r^2 - (o_1 - x)^2}$, $a_{i1} \leq x \leq a_{j1}$

(II) 在 B 号边上: 圆心坐标为 $O_b(l, \frac{b_{i2}+b_{j2}}{2})$, 曲线方程为 $x = o_1 - \sqrt{r^2 - (o_1 - y)^2}$, $b_{i2} \leq x \leq a_{j2}$

(III) 在 C 号边上: 圆心坐标 $O_c(\frac{c_{i1}+c_{j1}}{2}, 0)$, 曲线方程为 $y = o_2 - \sqrt{r^2 - (o_1 - x)^2}$, $c_{i1} \leq x \leq c_{j1}$

(IV) 在 D 号边上: 圆心坐标 $O_d(0, \frac{d_{i2}+d_{j2}}{2})$, 曲线方程为 $x = o_1 - \sqrt{r^2 - (o_1 - y)^2}$, $d_{i2} \leq y \leq d_{j2}$

3) 对于输入输出端口在对边且不平行的情况，从输出端引出的线路需要至少经过两次拐弯，才能以垂直于输入端所在边缘的方式与输入端相连，此时在该线路上不允许存在任何的交点。下面以任务三中唯一此类情况 (四号线路) 为例进行进一步阐述, 相关变量在下图中给出。

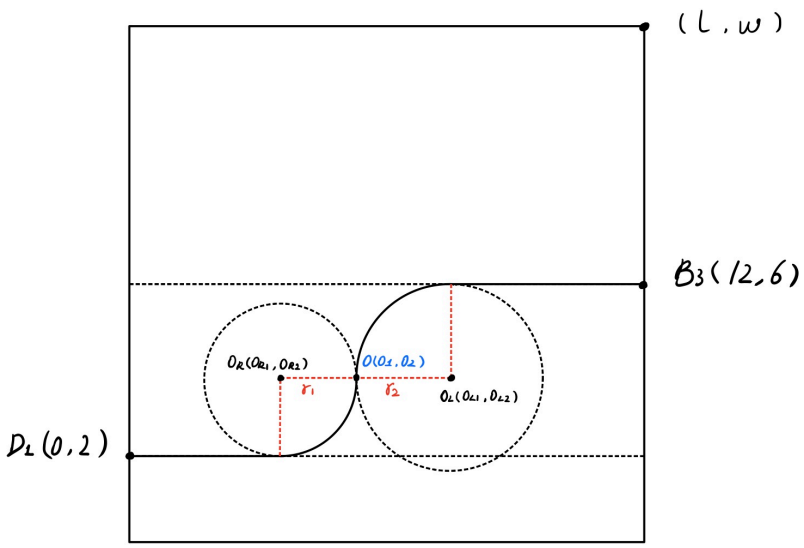


Figure 8

由于假设转弯半径尽可能大，转弯次数只有两次，在此类线路中，规定两次转弯半径和满足：$r_1+r_2=min\{|b_1-d_1|,|b_2-d_2|\}$ (此处线路板为正方形)。在满足这一条件下，由 D 到 B 之间的线路方程 $y_{BD}=f(x)$ 有界且 $m=min\{b_2,d_2\}\leq y_{BD}\leq max\{b_2,d_2\}=M$，此时，我们认为，平面 xoy 内任何不会进入两端点 B，D 所规定的矩形区域 $A_{BD}=\{(x,y)|\ 0\leq x\leq L, m\leq y\leq M\}$ 的曲线 (也即不同线路) 都不会和路径 BD 产生交点，也即他们可以同时存在在同一个线路板上。

综上所述，对两端口分别在 B，D 两边上的不平行情况，其优化曲线 BD 因满足以下几个特征：

(1) 对两次转弯半径和：$r_1+r_2=min\{|b_1-d_1|,|b_2-d_2|\}$

(2) 线路方程为中间段连续转两次 90 度的 “S 型” 方程且在端口确定的矩形区域 A_{BD} 内有界，其上下界即为 BD 线路函数在闭区间 $[0\ l]$ 的上下确界；

(3) 任何贯穿矩形区域 A_{BD} 的曲线都必会和 BD 有至少一个交点，任何不进入该区域的曲线 (包括在端点处相切) 都不会和曲线产生交点。
对于特征 (3)，定义线路板内任意连续的闭合曲线 $q(x,y)=0$“贯穿”A_{BD} 为能在线路板所划定的区域内同时取得 $q(x,m)=0$ $q(x,M)=0$，其中 m,M 分别为 BD 线路函数 $y_{BD}=f(x)$ 的最小，最大值。定义和特征描述的合理性可用闭区间内连续函数的零点存在性定理加以证明。下给出证明思路：

引理 5.3.1：若 $f(x)\in C[a,b], g(x)\in C[a,b]$ $Rangeg\subseteq Rangef$，则 $\exists\,\zeta\in[a,b]\ s.t.\ f(\zeta)=g(\zeta)$

证明：令 $h(x)=g(x)-f(x)\in C[a,b]$
设 M_1,M,m_1,m 分别为 g，f 的最大，最小
由于 g，f 均为闭区间内的连续函数，必有 $x_0\in[a,b]$ s.t. $g(x_0)=M_1$
所以若 $g(x_0)=M_1=M$，有 $h(x_0)=0, x_0\in[a,b]$
当 $g(x_1)=m_1=m$ 时同理

若 $g(x_0) \neq M$ 且 $g(x_1) \neq m$:
$h(x_0) = g(x_0) - f(x) = M_1 - f(x) > M - M_1 > 0$
$h(x_1) = g(x_1) - f(x) = m_1 - f(x) < m_1 - m < 0$
所以 $h(x_0)h(x_1) < 0$
有零点存在性定理 $\exists\, \zeta \in [a,b]\ f(\zeta) = g(\zeta)\}$ 即 f 和 g 必相交。
根据先前的讨论，路径 BD 在线路板规定区域的方程可以转化为一个闭区间连续函数 $f(x)$，而同一线路板上其他线路也都可以描述成有限区域内的连续方程。结合引理 5.3.1，可以证明特征 (3) 的合理性。

以两次拐弯轨迹的交点 (o_1,o_2) 为变量进一步考虑路径 BD 应满足的几何限制：

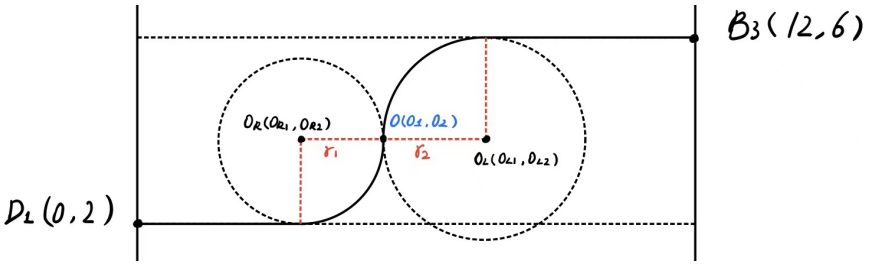


Figure 9

(4) 拐弯半径长度限制：题干要求拐弯半径不得小于 $1cm$，故有不等式组:

$$\begin{cases} r_1 = o_2 - d_2 \geq 1 \\ r_2 = b_2 - o_2 \geq 1 \end{cases} \tag{6}$$

(5) 接出 (入) 角度限制：接出点和接入点位置路径曲线 BD 的斜率为 0：

$$\begin{cases} o_{L1} = o_1 - r_1 = o_1 - o_2 + d_2 \geq 0 \\ o_{R1} = o_1 + r_2 = o_1 = b_2 - o_2 \leq Length \end{cases} \tag{7}$$

(6) 自然限制：路径的拐点应满足:

$$\begin{cases} 1 \geq o_1 \leq length - 1 \\ 1 \geq o_2 \leq width - 1 \end{cases} \tag{8}$$

综合上述特征 (1) (3) 与限制 (4) (6), 考虑在任务三条件下线路 4 的最优排布方式。由特征 (3) 知，在所有可能与线路 4 产生交互的线路中，仅有线路 2 可能与线路 4 的相交状况可能随不同的 (o_1,o_2) 而发生改变。且分析得知，若要使两线路相距较开，因尽量选取较小的 o_1 和较大的 o_2, 这可以通过改变两线路相切状态下线路 4 的特定参数值，观察两线路交点个数的变化看出。

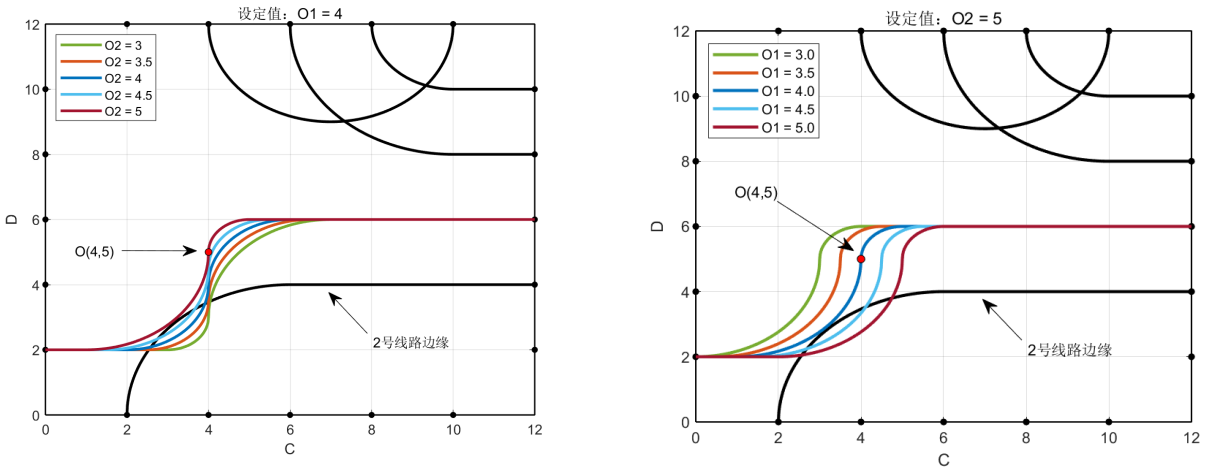


Figure 10: 单坐标固定的相交灵敏度分析

为获得能使线路 4 与 2 相距最远的参数 (o_1,o_2), 设定加权评价参数:$z = 0.8o_1 - 0.8o_2$ 并构建以其值最小为目标的线性规划模型：$minz = 0.8o_1 - 0.8o_2$

$$s.t.\begin{cases} 3 \le o_2 \le 5 \\ 1 \le o_1 \le 11 \\ o_2 \le o_1 + 2 \\ o_2 \ge o_1 - 6 \end{cases} \tag{9}$$

使用图解法求得最优解 $o_1 = 1, o_2 = 3, min\ z = 0.2$(见图 11)

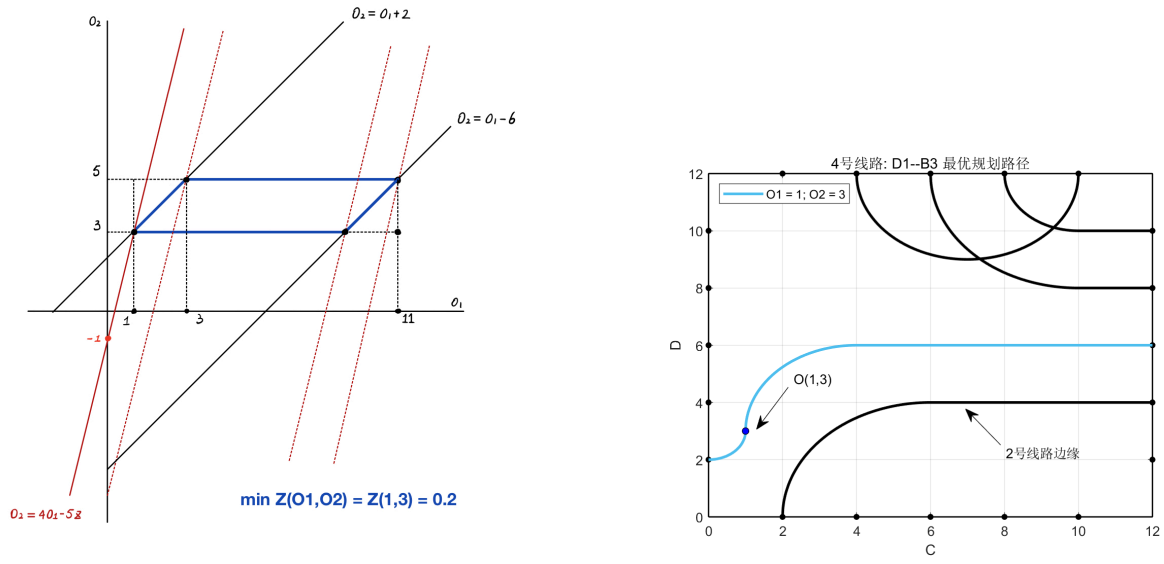


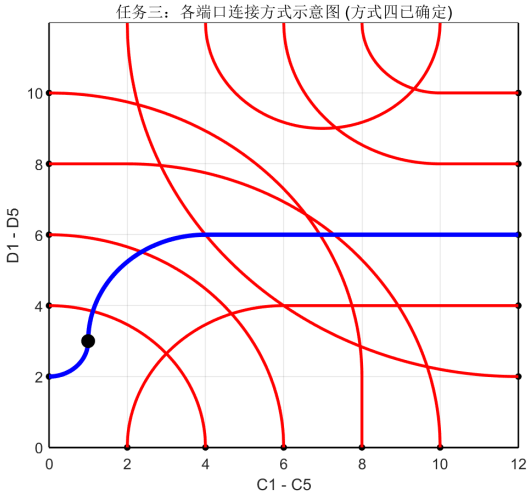
Figure 11: 最优线路 4

此时线路 4 的对应函数为

$$y=\begin{cases}3-\sqrt{1-x^2} & 0\le x\le 1\\ 3+\sqrt{9-(x-4)^2} & 1<x\le 4\\ 6 & 4<x\le 12\end{cases} \tag{10}$$

5.3.3 模型的建立

截至目前为止，我们对模型 3 中所有线路都给出了唯一确定的优化线路，有相应叠加线路图与排斥矩阵如下：



0	1	0	1	0	0	0	0	2	2
1	0	0	0	0	0	1	1	1	1
0	0	0	0	1	1	0	0	0	0
1	0	0	0	0	0	1	1	1	1
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0
2	1	0	1	0	0	0	0	0	1
2	1	0	1	0	0	0	0	1	0

Figure 12: 任务 3 叠加线路图及对应排斥矩阵

类似任务 2，保持目标函数，决策变量与其他限制条件不变，将 2 中条件 (3) 改为：

$$r_i m=\sum_{n=1}^{10} x_{im}x_{in}r_{mn}+x_{im}t_m\le 2$$

表示在任意一块板 i 上，导线转弯的次数和交叉数之和不大于 2。有规划模型：

$$min\ N$$

$$s.t.\begin{cases}x_j{}^{(N_t)}=\sum_{i=1}^{N_t}x_{ij}\le 1\\ \mathrm{r}_i m=\sum_{n=1}^{10}x_{im}x_{in}r_{mn}+x_{im}t_m\le 2,\ \mathrm{m}{=}1{\sim}10\\ x_j{}^{(N)}=1\end{cases} \tag{11}$$

5.3.4 模型的求解算法

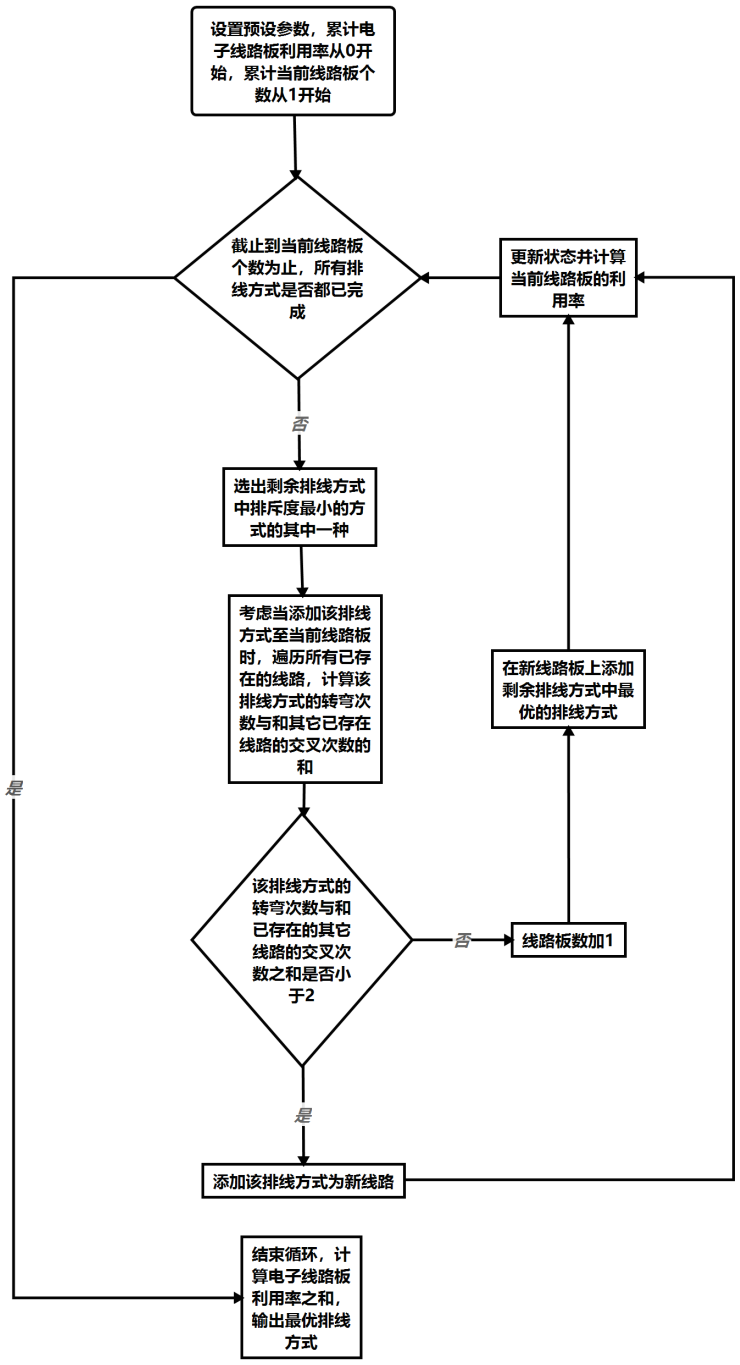


Figure 13: 流程图 3

使用 matlab 求得 $N = 3$。第一块板上选择 5，6，7，8，9，10 号线路进行连接。第二块板上选择 2，3，4 号线路连接。第三块板上只选择 1 号线路连接。各板利用率分别为 0.6,0.3，0.1。具体结果如下图：

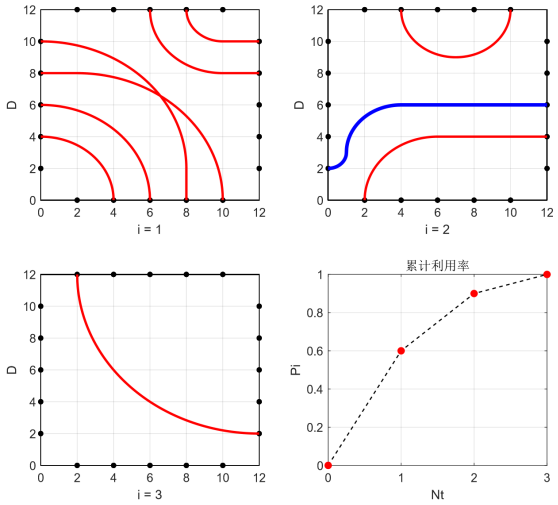


Figure 14: 任务 3 结果

5.4 任务 4：模型在 $18cm \times 18cm$ 规格线路板中的应用与求解

由于任务 4 中，只是电子线路板的端口变多，其整体的建模思路并没有变化，所以我们沿用任务 1 到 3 的模型，将新的端口连接方式应用至已建立的模型。现设置新的路线编号如下：

线路编号	1	2	3	4	5	6	7	8
起始端口	A1	B1	B2	A2	B4	A3	B5	B6
终止端口	B3	C2	C3	D2	C1	D1	C4	C5
线路编号	9	10	11	12	13	14	15	16
起始端口	A4	A5	B7	A6	A7	B8	D4	C8
终止端口	A8	C6	D5	D3	D6	C7	D8	D7

Table 4

5.4.1 $18cm \times 18cm$ 规格线路板中的不相交布线模型求解

叠加线路图与排斥矩阵：

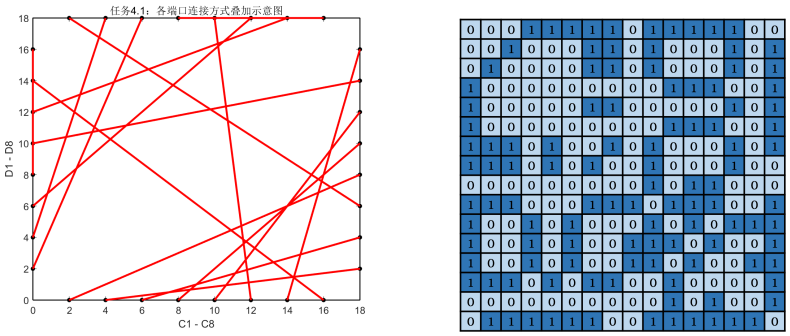


Figure 15: 任务 5.4.1 叠加线路图及对应排斥矩阵

求解结果如下图：

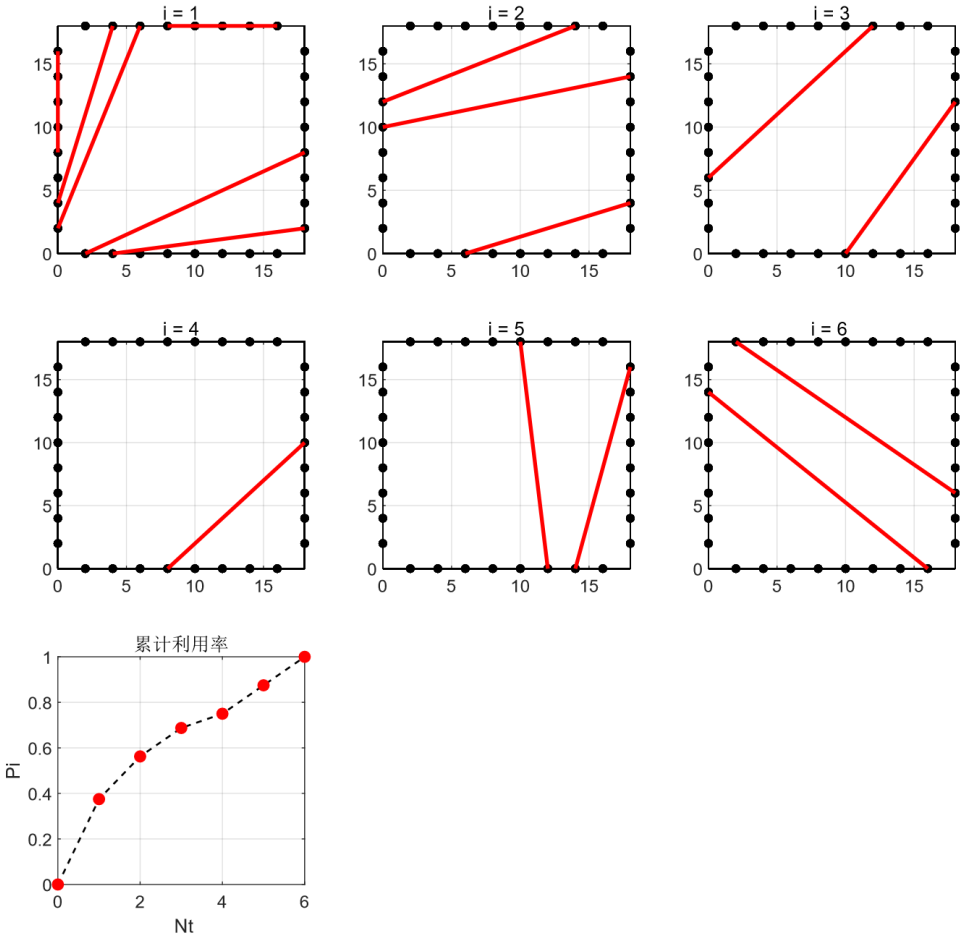


Figure 16: 任务 5.4.1 结果

由图可见，最少应使用 6 个相同的线路板，具体结果在下表中给出：

线路板层数	连接线路编号	单层线路板利用率
1	2,4,6,9,15	0.3750
2	3,11,13	0.1875
3	8,12	0.1250
4	7	0.0625
5	10,14	0.1250
6	1,16	0.1250

Table 5

5.4.2 18×18 规格线路板中的可相交布线模型求解

叠加线路图与排斥矩阵如 5.4.1

求解结果如下图：

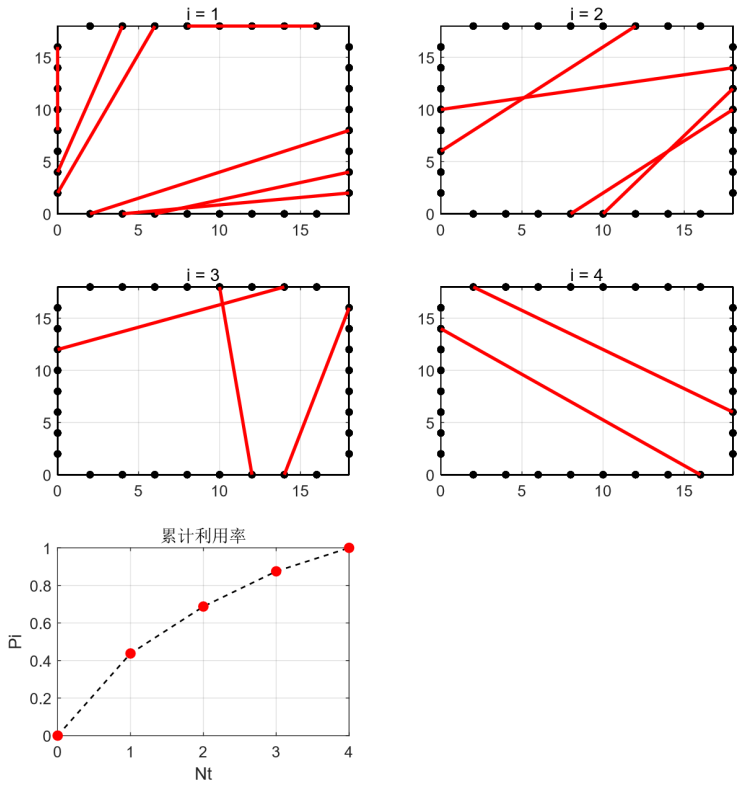


Figure 17: 任务 5.4.2 结果

由图可见，最少应使用 4 个相同的线路板，具体结果在下表中给出：

线路板层数	连接线路编号	单层线路板利用率
1	2,3,4,5,6,9,15	0.4375
2	7,8,11,12	0.2500
3	10,13,14	0.1875
4	1,16	0.1250

Table 6

5.4.3 18×18 规格线路板中的拐弯布线模型求解

注意这里有两种线路属于端点“对边不平行”的情况，分别为线路 10 与 11 号。11 号线路端口的相对位置和任务三中相似。根据此类线路的 (3) 号特征可以得出，和线路 11 相交状况可变的线路只有线路 8 和线路 13。

结合线路二次拐弯坐标 (o_1,o_2) 的几何限制，得出两参应满足：

$$\begin{cases} 1 \le o_1 \le 17 \\ 11 \le o_2 \le 13 \\ o_1 - 4 \le o_2 \le o_1 + 10 \end{cases} \tag{12}$$

考虑 o_2 分别取最值时线路 11 和线路 8，13 相切的临界情况，有

o_2 取值	o_1 的安全取值范围
13	$[11.9, 14.5]$
11	$[10.92, 13.4]$

Table 7

综合得出坐标 (o_1,o_2) 的安全取值区域为：

$$A_{safe} = \{(o_1, o_2) | \ 11.9 < o_1 < 13.4, 11 \leq o_2 \leq 13\}$$

在此区域内的 (o_1,o_2) 都能确保 11 号线路不会和线路 8，13 产生额外的交点，在没后续计算中，我们任取 $(o_1, o_2) = (12.26.11)$，同理考虑 10 号线路的二次拐弯点，设置为 $(11, 1)$。具体情况可由下图看出：

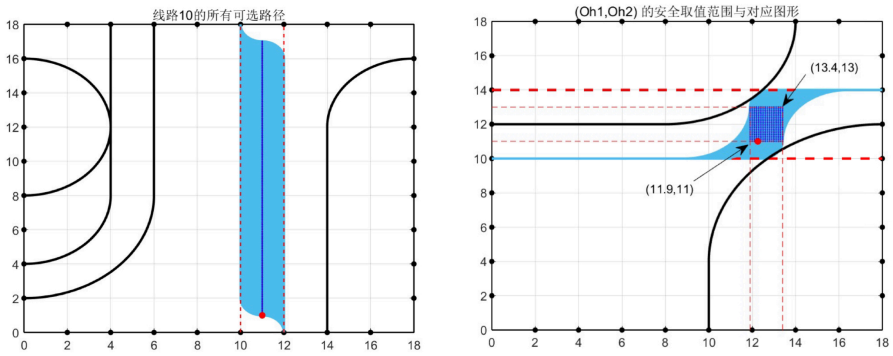


Figure 18: 线路 10 与 11 二次拐弯点的安全区间和对应线路图像

下面给出设定线路 10 和 11 二次拐弯点坐标分别为 (12.26.11),(11, 1) 下的线路叠加图和排斥矩阵。在这里认为多线相交于特定线路同一点时，该线路上的交叉点个数只计算一次，且其在排斥矩阵上对应位置的值设置为：

$$r_{mi} = \frac{1}{k}, \ i = n_1, ..., n_k (\text{线路 m 与 } n_{1k} \text{ 交于同一点})$$

给出叠加线路图与排斥矩阵为：

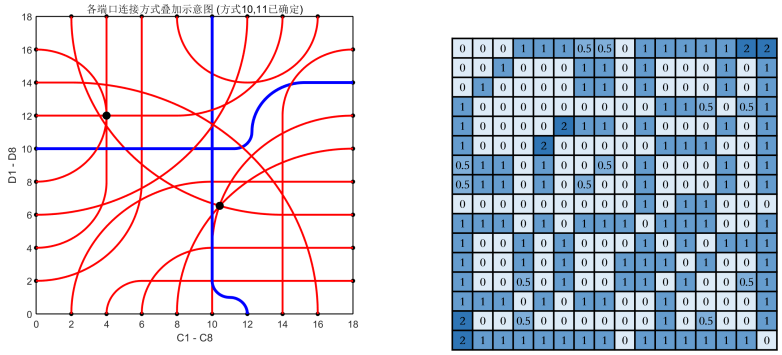


Figure 19: 任务 5.4.3 叠加线路图及对应排斥矩阵

求解结果如下图：

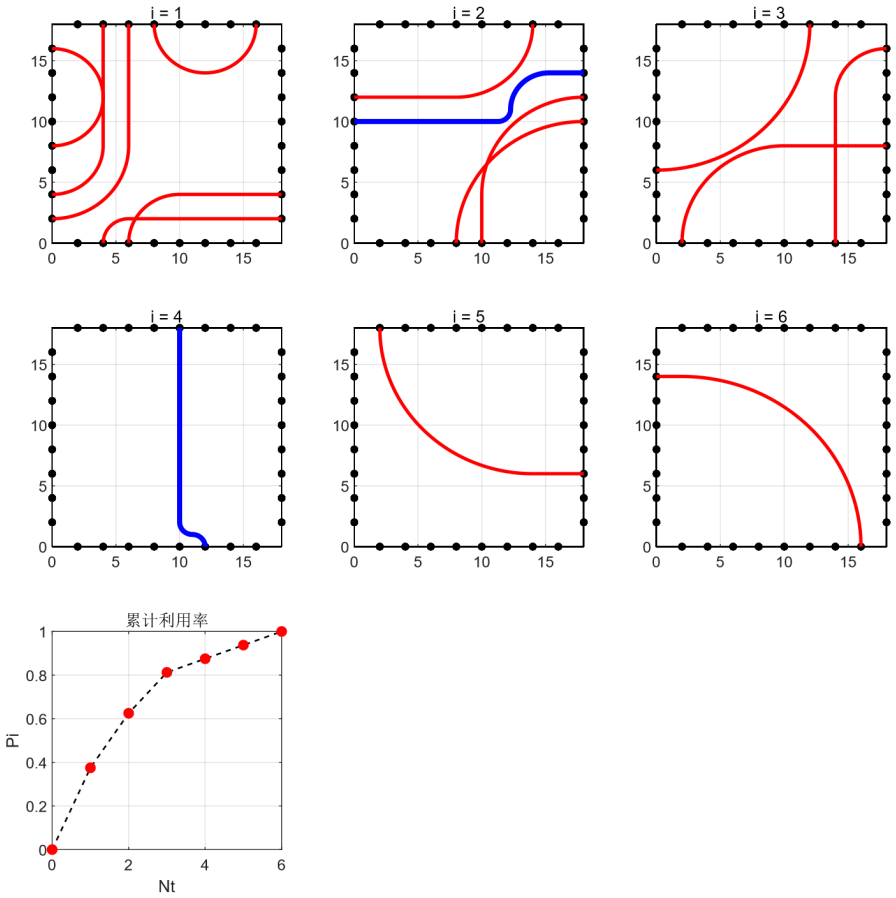


Figure 20: 任务 5.4.3 结果

由图可见，最少应使用 6 个相同的线路板，具体结果在下表中给出：

线路板层数	连接线路编号	单层线路板利用率
1	2,3,4,6,9,15	0.3750
2	7,8,11,13	0.2500
3	5,12,14	0.1875
4	10	0.0625
5	1	0.0625
6	16	0.0625

Table 8

6 模型的评价

6.1 模型的优点

1、使用 0-1 规划的思想抽象化布线问题，便于编程求解。

2、在已知各线路相交情况的前提下 (排斥矩阵已知)，采用贪心算法可以极大程度上减少计算量，提高解模速度。

3、模型求出的解均符合直观想象，一定程度上体现出模型的合理性与实用性。

4、对于不同限制条件的情况，模型总体改动较小，体现了该模型在实际应用中的灵活性。

5、提出了以相对位置为基础的最优线路规划，将不确定线路问题确定化，极大程度上节省了分析时间，且有较高的普适性。

6.2 模型的缺点

1、在线路相交情况 (排斥矩阵) 未知或难以分析时，模型将失效。

2、采用了启发式算法中最简单的贪心算法，容易陷入局部最优解。

3、贪心算法中评价线路优劣的标准比较单一，考虑不够全面；当唯一评价标准不清晰时容易导致算法失效。

4、拐弯模型中提供选取的优化线路在实际中也许并不一定为最优路径，这也在一定程度上降低了求出全局最优解的可能性。

6.3 模型的改进

1、针对缺陷 1，3，可以考虑同时使用其他用于判断线路选取优劣的参数进行加权综合评价，增加模型算法的适用场景。

2、针对缺陷 2，可以综合使用其他的启发式算法 (比如模拟退火) 以加强全局最优解的搜索能力。

3、针对缺陷 4，可以丰富优化线路的选取策略，提高算法求解的合理性。

附录:

(1) 5.3.1 **中** BC,CD,AD **邻边的公式**

BC: $r = |c_2 - b_2|,\ O_{bc}\ (c_1 + r, c_2)$

$$\begin{cases} y = o_2 + \sqrt{r^2 - (o_1 - x)^2} & ,\ c_1 \le x \le o_1 \\ y = b_2 & ,\ \mathrm{o_1{<}x} \le b_1 \end{cases} \tag{13}$$

$r = |c_1 - b_1|,\ O_{bc}\ (b_1, b_2 - r)$

$$\begin{cases} x = c_1 & ,\ \mathrm{c_2{<}y} \le o_2 \\ x = o_1 - \sqrt{r^2 - (o_2 - y)^2} & ,\ o_2 \le y \le b_2 \end{cases} \tag{14}$$

CD: $r = |c_2 - d_2|,\ O_{cd}\ (c_1 - r, c_2)$

$$\begin{cases} y = d_2 & ,\ 0 < x \le o_1 \\ y = o_2 + \sqrt{r^2 - (o_1 - x)^2} & ,\ o_1 \le x \le c_1 \end{cases} \tag{15}$$

$r = |c_1 - d_1|,\ O_{cd}\ (d_1, d_2 - r)$

$$\begin{cases} x = c_1 & ,\ 0 < y \le o_2 \\ x = o_1 + \sqrt{r^2 - (o_2 - y)^2} & ,\ o_2 \le y \le d_2 \end{cases} \tag{16}$$

AD: $r = |a_2 - d_2|,\ O_{ad}\ (a_1 - r, a_2)$

$$\begin{cases} y = d_2 & ,\ 0 < x \le o_1 \\ y = o_2 - \sqrt{r^2 - (o_1 - x)^2} & ,\ o_1 \le x \le a_1 \end{cases} \tag{17}$$

$r = |a_1 - d_1|,\ O_{ad}\ (d_1, d_2 + r)$

$$\begin{cases} x = o_1 + \sqrt{r^2 - (o_2 - y)^2} & ,\ d_2 \le y \le o_2 \\ x = a_1 & ,\ \mathrm{o_2{<}y} \le a_2 \end{cases} \tag{18}$$

(2) **代码实现** (所有代码基于 matlab2019b)

任务 1

```
%% main
global FinalResult leng N cum
clc;clear;
leng = 12;
% 导入排斥矩阵
R = [0,1,0,1,0,0,0,0,0,0;
     0,0,0,0,0,0,1,1,1,1;
     0,0,0,0,1,1,0,0,0,0;
     0,0,0,0,0,0,1,1,1,1;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,1;
     0,0,0,0,0,0,0,0,0,0;];
R = R'+R; r = sum(R);


% 设置迭代参数
Nt = 1; Num = 10;
p = []; X = zeros([1,Num]);
isend = 0; tempr1 = r;
% 开始迭代
while isend == 0
    tempr2 = tempr1;
    for i = 1:length(find(tempr1~=Inf))  % 寻找当前最优可选路线
        J = find(tempr2 == min(tempr2));
        j = J(1);

        addble = add(j,X,R); % 判断路线是否可以添加到当前线路板
        if addble
            X(Nt,j) = 1; % 更新当前线路板状态
            tempr1(j) = inf;
        break
        else
            tempr2(j) = inf;
        end
    end

    if sum(tempr2 == inf) == Num  % 当没有可选路线时，添加新线路板
        p(Nt) = sum(X(Nt,:))/Num; % 计算当前线路板的利用率
        Nt = Nt + 1;
        X = [X;zeros([1,Num])];
```

```
        J = find(tempr1 == min(tempr1));
        j = J(1);
        X(Nt,j) = 1;
        tempr1(j) = inf;
    end

    [isend,N] = endWhile(X);   % 判断是否所有路线都出现过至少一次

    if isend
        p(N) = sum(X(N,:))/Num;
    end
end


% 输出所需参数
N; cum = cumsum(p); FinalResult = X;
```

```
%% 判断线路是否满足可加条件
function result = add(j,X,R)
length = size(X,2); xi = X(end,:);
xi(j) = 1; ri = 0;
for m = 1:length
    for n = 1:length
        ri = ri + prod([xi(m),xi(n),R(m,n)]);
    end
end
if ri == 0
    result = 1;
else
    result = 0;
end
end
```

```
%% 判断布局是否满足结束条件
function [isend,N] = endWhile(X)
x = sum(X);
if sum(x) == 10
    isend = 1;
    N = size(X,1);
else
    isend = 0;
    N = 0;
end
end
```

任务 2(主函数和可视化代码与任务 1 相同)

```
%% 判断线路是否满足可加条件
function addble = add(j,X,R)
length = size(X,2);
Result = zeros([1,length]);
xi = X(end,:);
xi(j) = 1;
for m = 1:length
    rim = 0;
    for n = 1:length
        rim = rim + prod([xi(m),xi(n),R(m,n)]);
    end
    Result(m) = rim;
end
Det = (Result > 1);
if sum(Det) == 0
    addble = 1;
else
    addble = 0;
end
end
```

任务 3

```
%% main
clc;clear;
global FinalResult N
leng = 12;
% 导入排斥矩阵
R = [0,1,0,1,0,0,0,0,2,2;
     0,0,0,0,0,0,1,1,1,1;
     0,0,0,0,1,1,0,0,0,0;
     0,0,0,0,0,0,1,1,1,1;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,0;
     0,0,0,0,0,0,0,0,0,1;
     0,0,0,0,0,0,0,0,0,0;];
R = R'+R; r = sum(R);

% 设置每条线路对应转弯次数
```

```
T = ones([1,10]); T(4) = 2;

% 设置迭代参数
Nt = 1; Num = 10;
p = []; X = zeros([1,Num]);
isend = 0; tempr1 = r;

% 开始迭代
while isend == 0
    tempr2 = tempr1;
    for i = 1:length(find(tempr1~=Inf))  % 寻找当前最优可选路线
        J = find(tempr2 == min(tempr2));
        j = J(1);

        addble = add(j,X,R,T); % 判断路线是否可以添加到当前线路板
        if addble
            X(Nt,j) = 1; % 更新当前线路板状态
            tempr1(j) = inf;
        break
        else
            tempr2(j) = inf;
        end
    end

    if sum(tempr2 == inf) == Num  % 当没有可选路线时，添加新线路板
        p(Nt) = sum(X(Nt,:))/Num; % 计算当前线路板的利用率
        Nt = Nt + 1;
        X = [X;zeros([1,Num])];
        J = find(tempr1 == min(tempr1));
        j = J(1);
        X(Nt,j) = 1;
        tempr1(j) = inf;
    end

    [isend,N] = endWhile(X);
    if isend
        p(N) = sum(X(N,:))/Num;
    end
end


% 输出所需参数
N; cum = cumsum(p); FinalResult = X;

%% 判断线路是否满足可加条件
```

```
function addble = add(j,X,R,T)
length = size(X,2);
Result = zeros([1,length]);
xi = X(end,:);
xi(j) = 1;


for m = 1:length
    rim = 0;
    for n = 1:length
        rim = rim + prod([xi(m),xi(n),R(m,n)]);
    end
    rim = rim + xi(m)*T(m);
    Result(m) = rim;
End

Det = (Result > 2);
if sum(Det) == 0
    addble = 1;
else
    addble = 0;
end
end
```

```
%% 判断是否满足终止条件
function [isend,N] = endWhile(X)
x = sum(X);
if sum(x) == 10
    isend = 1;
    N = size(X,1);
else
    isend = 0;
    N = 0;
end
end
```

可视化任务 1,2

```
%% 可视化任务 1,2
global leng NumOfPorts
NumOfPorts = 5; Num = 10;

% 输入 A~D 边上端口的坐标信息
XA = (2:2:NumOfPorts*2)'; YA = leng.*ones([NumOfPorts,1]);
```

```
XB = YA; YB = XA;
XC = XA; YC = zeros([NumOfPorts,1]);
XD = YC; YD = XA;
global A B C D
A = [XA,YA]; C = [XC,YC];
B = [XB,YB]; D = [XD,YD];

% 输入线路信息和输入输出端口相对位置
global method cate
cate = [2,2,3,4,2,2,2,2,2,2]; % {1,2,3,4} 对边平行，邻边，同边，对边不平行
method = cell(1,10);
method{1} = [A(1,:);B(1,:)]; method{2} = [B(2,:);C(1,:)];
method{3} = [A(2,:);A(5,:)]; method{4} = [D(1,:);B(3,:)];
method{5} = [B(4,:);A(3,:)]; method{6} = [A(4,:);B(5,:)];
method{7} = [C(2,:);D(2,:)]; method{8} = [D(3,:);C(3,:)];
method{9} = [C(4,:);D(5,:)]; method{10} = [C(5,:);D(4,:)];

% plot Frame
Frame = [0,0; 0,leng;leng, leng; leng,0;0,0];
plotframe(Frame);

% 绘制线路叠加图
for i = 1:10
 drawline(i)
end
```

```
%% 可视化最终结果
global FinalResult N leng cum
% 绘制每层板的线路连接情况
for i = 1:N
    arrange = FinalResult(i,:);
    showplot = find(arrange==1);

    % plot Frame
    Frame = [0,0; 0,leng;leng, leng; leng,0;0,0];
    subplot(2,2,i),
    plotframe(Frame);
    xlim([0,leng]); ylim([0,leng]);
    grid on

    for j = showplot
        drawline(j)
    end
    hold off
```

```
 end


% 绘制累计利用率曲线
subplot(2,2,4),
plot((0:N)',[0,cum],"k--","LineWidth",1)
xlim([0,N]); ylim([0,1]);
grid on
hold on
plot((0:N)',[0,cum],"r.","MarkerSize",20)
```

可视化任务 3

```
%% 可视化任务 3
global leng NumOfPorts
NumOfPorts = 5;
Num = 10;

% 输入 A~D 边上端口的坐标信息
XA = (2:2:NumOfPorts*2)'; YA = leng.*ones([NumOfPorts,1]);
XB = YA; YB = XA;
XC = XA; YC = zeros([NumOfPorts,1]);
XD = YC; YD = XA;
global A B C D
A = [XA,YA]; C = [XC,YC];
B = [XB,YB]; D = [XD,YD];

% 输入线路信息和输入输出端口相对位置
global method cate
cate = [2,2,3,4,2,2,2,2,2,2]; % {1,2,3,4} 对边平行，邻边，同边，对边不平行
method = cell(1,10);
method{1} = [A(1,:);B(1,:)]; method{2} = [B(2,:);C(1,:)];
method{3} = [A(2,:);A(5,:)]; method{4} = [D(1,:);B(3,:)];
method{5} = [B(4,:);A(3,:)]; method{6} = [A(4,:);B(5,:)];
method{7} = [C(2,:);D(2,:)]; method{8} = [D(3,:);C(3,:)];
method{9} = [C(4,:);D(5,:)]; method{10} = [C(5,:);D(4,:)];

% plot Frame
Frame = [0,0; 0,leng;leng, leng; leng,0;0,0];
plotframe(Frame);


% 绘制线路叠加图
for i = 1:10
 drawline(i)
```

```
end
```

```
%% 可视化最终结果
global FinalResult N leng p
% 绘制每层板的线路连接情况
for i = 1:N
    arrange = FinalResult(i,:);
    showplot = find(arrange==1);

    % plot Frame
    Frame = [0,0; 0,leng;leng, leng; leng,0;0,0];
    subplot(2,2,i),
    plotframe(Frame);
    xlim([0,leng]); ylim([0,leng]);
    grid on

    for j = showplot
        if j ~= 4
            drawplot(j)
        else  % 绘制对边情况
            y4 = @(x)type4(x,1,3);
            x = (0:0.01:12)';
            plot(x,y4(x),"b-","LineWidth",3)
        end
    end
    hold off
end


% 绘制累计利用率曲线
subplot(2,2,4),
cum = cumsum(p);
plot((0:N)',[0,cum],"k--","LineWidth",1)
xlim([0,N]); ylim([0,1]);
grid on
hold on
plot((0:N)',[0,cum],"r.","MarkerSize",20)
```

```
%% 计算不同相对位置下的曲线坐标
function Coor = draw(target)
global cate method leng
targetcate = cate(target);
startx = method{target}(1,1); starty = method{target}(1,2);
```

```
endx = method{target}(2,1)   ; endy = method{target}(2,2);


switch targetcate
    case 1 % 对边平行
        if starty == endy
            X = (0:0.01:leng)';
            Y = startx*ones([size(X,1),1]);

        elseif startx == endx
            Y = (0:0.01:leng)';
            X = startx*ones([size(Y,1),1]);
        end


    case 2 % 邻边
        % AB
        if (starty == leng && endx == leng) ...
           ||(startx == leng && endy == leng)
           if starty == leng
               a = method{target}(1,:);
               b = method{target}(2,:);
           else
               a = method{target}(2,:);
               b = method{target}(1,:);
           end

           r = min(abs(a(1)-b(1)),abs(a(2)-b(2)));
           if r == abs(a(2)-b(2))
               o = [a(1)+r,a(2)];
               X1 = (a(1):0.01:o(1))'; Y1 = o(2)-sqrt(r^2-(o(1)-X1).^2);
               X2 = (o(1):0.01:b(1))'; Y2 = b(2)*ones([size(X2,1),1]);
               X = [X1;X2]; Y = [Y1;Y2];


           elseif r == abs(a(1)-b(1))
               o = [b(1),b(2)-r];
               Y1 = (b(2):0.01:o(2))'; X1 = o(1)-sqrt(r^2-(o(2)-Y1).^2);
               Y2 = (o(2):0.01:a(2))'; X2 = a(1)*ones([size(Y2,1),1]);
               X = [X1;X2]; Y = [Y1;Y2];
           end

        % AD
        elseif (starty == leng && endx == 0)...
           ||(startx == 0 && endy == leng)
```

```
        if starty == leng
            a = method{target}(1,:);
            d = method{target}(2,:);
        else
            a = method{target}(2,:);
            d = method{target}(1,:);
        end

        r = min(abs(a(1)-d(1)),abs(a(2)-d(2)));
        if r == abs(a(2)-d(2))
            o = [a(1)-r,a(2)];
            X1 = (0:0.01:o(1))'; Y1 = d(2)*ones([size(X1,1),1]);
            X2 = (o(1):0.01:a(1))'; Y2 = o(2)-sqrt(r^2-(o(1)-X2).^2) ;
            X = [X1;X2]; Y = [Y1;Y2];


        elseif r == abs(a(1)-d(1))
            o = [d(1),d(2)+r];
            Y1 = (d(2):0.01:o(2))'; X1 = o(1)+sqrt(r^2-(o(2)-Y1).^2);
            Y2 = (o(2):0.01:a(2))'; X2 = a(1)*ones([size(Y2,1),1]);
            X = [X1;X2]; Y = [Y1;Y2];
        end

    % CD
    elseif (starty == 0 && endx == 0)...
        ||(startx == 0 && endy == 0)
        if starty == 0
            c = method{target}(1,:);
            d = method{target}(2,:);
        else
            c = method{target}(2,:);
            d = method{target}(1,:);
        end

        r = min(abs(c(1)-d(1)),abs(c(2)-d(2)));
        if r == abs(c(2)-d(2))
            o = [c(1)-r,c(2)];
            X1 = (0:0.01:o(1))'; Y1 = d(2)*ones([size(X1,1),1]);
            X2 = (o(1):0.01:c(1))'; Y2 = o(2)+sqrt(r^2-(o(1)-X2).^2) ;
            X = [X1;X2]; Y = [Y1;Y2];


        elseif r == abs(c(1)-d(1))
            o = [d(1),d(2)-r];
```

```
            Y1 = (0:0.01:o(2))'; X1 = c(1)*ones([size(Y1,1),1]);
            Y2 = (o(2):0.01:d(2))'; X2 = o(1)+sqrt(r^2-(o(2)-Y2).^2);
            X = [X1;X2]; Y = [Y1;Y2];
        end

    % CB
    elseif (starty == 0 && endx == leng)...
        ||(startx == leng && endy == 0)
        if starty == 0
            c = method{target}(1,:);
            b = method{target}(2,:);
        else
            c = method{target}(2,:);
            b = method{target}(1,:);
        end

        r = min(abs(c(2)-b(2)),abs(c(1)-b(1)));
        if r == abs(c(2)-b(2))
            o = [c(1)+r,c(2)];
            X1 = (c(1):0.01:o(1))'; Y1 = o(2)+sqrt(r^2-(o(1)-X1).^2);
            X2 = (o(1):0.01:b(1))'; Y2 = b(2)*ones([size(X2,1),1]);
            X = [X1;X2]; Y = [Y1;Y2];

        elseif r == abs(c(1)-b(1))
            o = [b(1),b(2)-r];
            Y1 = (c(2):0.01:o(2))'; X1 = c(1)*ones([size(Y1,1),1]);
            Y2 = (o(2):0.01:b(2))'; X2 = o(1)-sqrt(r^2-(o(2)-Y2).^2);
            X = [X1;X2]; Y = [Y1;Y2];
        end
    end

case 3 % 同边
    % A
    if starty == endy && endy == leng
        ai = method{target}(1,:);
        aj = method{target}(2,:);
        o = [(ai(1)+aj(1))/2,leng];
        x1 = min([ai(1),aj(1)]);
        x2 = max([ai(1),aj(1)]);
        r = (ai(1)+aj(1))/2 - x1;

        X = (x1:0.01:x2)';
        Y = o(2)-sqrt(r^2-(o(1)-X).^2);
```

```
        % B
        elseif startx == endx && endx == leng
            bi = method{target}(1,:);
            bj = method{target}(2,:);
            o = [leng,(bi(2)+bj(2))/2];
            y1 = min([bi(2),bj(2)]);
            y2 = max([bi(2),bj(2)]);
            r = (bi(2)+bj(2))/2 - y1;

            Y = (y1:0.01:y2)';
            X = o(1)-sqrt(r^2-(o(2)-Y).^2);
        end

        % C
        if starty == endy && endy == 0
            ci = method{target}(1,:);
            cj = method{target}(2,:);
            o = [(ci(1)+cj(1))/2,0];
            x1 = min([ci(1),cj(1)]);
            x2 = max([ci(1),cj(1)]);
            r = (ci(1)+cj(1))/2 - x1;

            X = (x1:0.01:x2)';
            Y = o(2)+sqrt(r^2-(o(1)-X).^2);

        % D
        elseif startx == endx && endx == 0
            di = method{target}(1,:);
            dj = method{target}(2,:);
            o = [0,(di(2)+dj(2))/2];
            y1 = min([di(2),dj(2)]);
            y2 = max([di(2),dj(2)]);
            r = (di(2)+dj(2))/2 - y1;

            Y = (y1:0.01:y2)';
            X = o(1)+sqrt(r^2-(o(2)-Y).^2);
        end

    case 4 % 对边不平行
        X = [];
        Y = [];

end
Coor = [X,Y];
```

```
%% 绘制对边情况曲线
function f = plottype4(01,02,k)
y4 = @(x)typ4(x,01,02);
x = (0:0.01:12)';
switch k
    case "b"
        f = plot(x,y4(x),"-","LineWidth",2,"Color",'#0072BD')
    case "g"
        f = plot(x,y4(x),"-","LineWidth",2,"Color",'#77AC30')
    case "r"
        f = plot(x,y4(x),"-","LineWidth",2,"Color",'#A2142F')
    case "c"
        f = plot(x,y4(x),"-","LineWidth",2,"Color",'#4DBEEE')
    case "o"
        f = plot(x,y4(x),"-","LineWidth",2,"Color",'#D95319')
end
grid on
end
```

```
%% 绘制其他情况曲线
function drawplot(method)
Coor = draw(method);
if length(Coor) ~= 0
X = Coor(:,1);
Y = Coor(:,2);
plot(X,Y,"k-","LineWidth",2)
end
end
```

```
%% plotframe
function plotframe(Frame)
global leng
plot(Frame(:,1),Frame(:,2),"k-","LineWidth",1)
hold on
width = 2;
partition = (2:width:leng-2)';
m1 = leng * ones(size(partition,1));
m2 = zeros(size(partition,1));
plot(partition,m1,"ko","MarkerSize",4,"MarkerFaceColor","black")
plot(partition,m2,"ko","MarkerSize",4,"MarkerFaceColor","black")
plot(m1,partition,"ko","MarkerSize",4,"MarkerFaceColor","black")
plot(m2,partition,"ko","MarkerSize",4,"MarkerFaceColor","black")
end
```