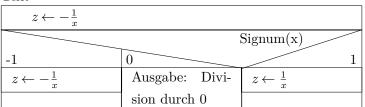
algorithm

n+e

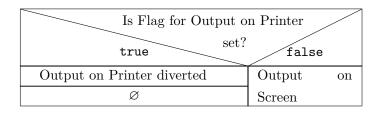
July 2017

1 Algorithm

Text



I	
I < 99	
	$J \leftarrow I + 1$
	Swap, if valid: $ARRAY(I) > ARRAY(J)$
	$J \leftarrow J + 1$
	J < 100
	$I \leftarrow I + 1$



```
Input: 当前芯片的布线状态, 其中靠近原点的子区域尚未布线
  Output: 在该子区域布线之后的芯片布线状态
1 while 子区域中连线起点尚未全部连接 / 有剩余连线终点 do
    if 已经到达芯片角落 then
      退出该循环;
3
    end
4
    if 连线终点落后于连线起点 then
5
      选择 n_x(n_y) 所在的那一列 (行), 左一个右一个地连出去,
       直到边界被连接的连线终点连续排列;
    end
7
    Call function Link_in to obtain pt_{x_1}, pt_{y_1}, 以及备选的两条线路;
8
    计算 f_x, f_y 的值,如果某个值为 1 则置该侧线路长度为 +\infty;
9
    if 某侧的 f 值不为 1 \land 计算该侧线路失败 then
10
    return 该子区域无合法布线状态;
11
    end
12
    比较两条线路长度,将长度小的一条记录至布线状态中;
13
    if 另一条线路的起点如果连向对侧的长度比该线路要长 then
14
     将该线路也记录至布线状态中;
15
    \mathbf{end}
16
17 end
18 if 子区域中连线起点尚未全部连接 / 无剩余连线终点 then
   return 该子区域无合法布线状态;
20 end
21 while 子区域中连线起点尚未全部连接 do
   if 选取某一列全部连接至 x 轴的总长度要比选取某一行全部连
    接至 y 轴的总长度要短 then
     选取 n_x 所在的一列,将其全部连接至 x 轴;
23
    else
24
     选取 n_y 所在的一行,将其全部连接至 y 轴;
    end
26
27 end
28 return 当前芯片的布线状态;
```

Algorithm 1: 子区域中的布线算法.

```
Input: The length node_x and width node_y of the biochip.
  Output: 最小的间隔 d、布线总长度和布线方案
1 if \min\{N, M\} \le 2 then
2 直接构造出方案并返回;
3 end
4 将连线起点划分成 4 个子区域;
5 确定 d 的上下界,使用二分法判定在该状态下是否有可行方案;
6 foreach 二分出来的 d do
7
     Calculating total length and width of biochip;
     Connecting origins near central axis;
8
     for 每个子区域 do
9
       Call Algorithm 1 to obtain 子区域中的布线方案 with 当前芯
10
        片布线状态 as input;
       if 该函数返回布线失败 then
11
          d 不合法,退出该循环;
12
       \quad \text{end} \quad
13
       绕坐标轴进行坐标变换;
14
     \mathbf{end}
15
     计算下一个需要判断的 d;
16
17 end
18 return 最小的间隔 d、布线总长度和布线方案;
```

Algorithm 2: Complete routing method of RRBI.????