

tikz-flowchart—TikZ 流程图绘制宏包^{*}

耿楠[†]

西北农林科技大学信息工程学院计算机科学系

2019/08/20

摘要

这是一个使用 TikZ 绘制传统程序流程图的简单宏包，通过定义 $\langle proc \rangle$ 、 $\langle test \rangle$ 、 $\langle io \rangle$ 、 $\langle term \rangle$ 等 TikZ 的 `\node` 命令的样式选项实现。该宏包核心代码摘录自 Brent Longborough 设计的流程图绘制样例，参考了 `tikz-imagelabels` 宏包的设计思路，提供了 `\flowchartset` 命令以设置绘制参数。

目录

1 宏包简介	2
2 使用方法	4
2.1 载入宏包	4
2.2 流程图绘制样式	4
2.3 样式参数设置命令	5
3 流程图绘制步骤	6
3.1 布置结点	6
3.2 布置坐标点	7

^{*}该文档是 `tikz-flowchart v1.0.01`, dated 2019/08/20 的说明文档。

[†]<https://github.com/registor/tikz-flowchart>

3.3	绘制流程线	7
3.4	绘制其它图形	8
4	流程图样例	8
4.1	改变流程线颜色	8
4.2	改变流程框填充色	8
4.3	局部设置	10
4.4	使用 TikZ 命令和参数	13
5	代码实现	13
5.1	宏包选项	13
5.2	配置命令	14
5.3	默认参数值	15
5.4	样式定义	15
5.5	调试命令定义	19

Change History

v1.0.01

General: chang .sty file to .dtx file . 1

1 宏包简介

流程图是诸如手册、报告、论文等文档中经常用到的排版元素, `tikz-flowchart` 宏包的目的是为了更为方便地实现传统流程图的绘制。图 1 是使用 `tikz-flowchart` 宏包绘制 `for` 循环结构的一个简单示例。

图 1 由如下代码绘制:

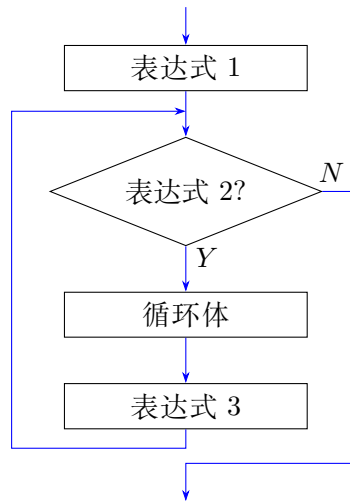


图 1: for 循环流程图

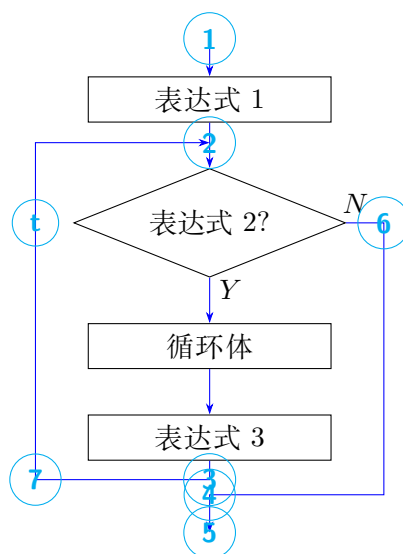
```

% \begin{tikzpicture}
%   % 布置结点
%   \node [proc] (p1) {表达式1};
%   \node [test, join] (t1) {表达式2?};
%   \node [proc] (p2) {循环体};
%   \node [proc, join] (p3) {表达式3};
%   % 布置用于连接的坐标结点，同时为其布置调试标记点。
%   \node [coord, above = 0.5 of p1] (c1) {};
%   \node [coord] (c2) at ($(p1)!0.35!(t1)$) {};
%   \node [coord, below = 0.25 of p3] (c3) {};
%   \node [coord, below = 0.2 of c3] (c4) {};
%   \node [coord, below = 0.5 of c4] (c5) {};
%   \node [coord, right = 0.5 of t1] (c6) {};
%   \node [coord, left = 0.5 of t1] (ct) {};
%   \node [coord] (c7) at (c3 -| ct) {};
%   % 判断框连线，每次绘制时，先绘制一个带有一个固定
%   % 位置标注的路径(path)，然后再绘制箭头本身(arrow)。
%   \path (t1.south) -- node [near start, right] {$Y$} (p2.north);
%   \draw [norm] (t1.south) -- (p2.north);
%   \path (t1.east) -- node [near start, above] {$N$} (c6);
%   \draw [norm] (t1.east) -- (c6) |- (c4) -- (c5);
%   % 其它连线

```

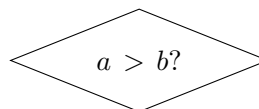
注意：所有绘图代码都需在 `tikzpicture` 环境中完成。

2.1 载人宏包



2.2 流程图绘制样式

Arg [\node] proc proc: \node 命令绘制顺序执行框的样式，如：



Arg [\node] test **test:** \node 命令绘制判断框的样式, 如:

Arg [\node] io **io:** \node 命令绘制输入/输出框的样式, 如:

Arg [\node] term **term:** \node 命令绘制开始/结束框的样式, 如:

Arg [\node] it **it:** \node 命令绘制斜体标注的样式, 如: 这里需要特别注意

Arg [\node] connector **connector:** \node 命令绘制流程图链接点的样式, 如:

Arg [\node] coord **coord:** \node 命令布置转角结点的样式, 如:

Arg [\draw] connect **connect:** \draw 命令绘制不相交的两个交汇路径的样式, 如:

Arg [\draw] lnorm **lnorm:** \draw 命令绘制指定颜色无箭头连线的样式, 如:

Arg [\draw] lfree **lfree:** \draw 命令绘制指定颜色无箭头连线的样式, 如:

Arg [\draw] lcong **lcong:** \draw 命令绘制指定颜色无箭头连线的样式, 如:

Arg [\draw] norm **norm:** \draw 命令绘制指定颜色箭头连线的样式, 如:

Arg [\draw] free **free:** \draw 命令绘制指定颜色箭头连线的样式, 如:

Arg [\draw] dotnorm **dotnorm:** \draw 命令绘制指定颜色流程线实心交点的样式, 如:

Arg [\draw] dotfree **dotfree:** \draw 命令绘制指定颜色流程线实心交点的样式, 如:

Arg [\draw] dotcong **dotcong:** \draw 命令绘制指定颜色流程线实心交点的样式, 如:

Arg [\draw] cdotnorm **cdotnorm:** \draw 命令绘制指定颜色流程线空心交点的样式, 如:

Arg [\draw] cdotfree **cdotfree:** \draw 命令绘制指定颜色流程线空心交点的样式, 如:

Arg [\draw] cdotcong **cdotcong:** \draw 命令绘制指定颜色流程线空心交点的样式, 如:

2.3 样式参数设置命令

\flowchartset tikz-flowchart 宏包定义了\flowchartset 命令, 该命令用于设置流程图各元

素的的绘图属性，详情见如下代码注释。

```
\flowchartset{
    free color = green,           % 自由连线颜色（默认取 green）
    norm color = blue,           % 常规连线颜色（默认取 blue）
    cong color = red,            % 关联连线颜色（默认取 red）
    proc fill color = white,      % 顺序处理框填充颜色（默认取白色）
    test fill color = white,      % 判断框填充颜色（默认取白色）
    io fill color = white,        % 输入/输出框填充颜色（默认取白色）
    term fill color = white,      % 开始/结束框填充颜色（默认取白色）
    proc text width = 8em,        % 顺序处理框宽度（默认取 8em）
    test text width = 5em,        % 判断框宽度（默认取 5em）
    io text width = 6em,          % 输入/输出框宽度（默认取 6em）
    term text width = 3em,        % 开始/结束宽度（默认取 3em）
    chain direction = below,      % 结点自动布置方向（默认取 below）
    minimum node distance = 6mm,  % 最小结点间距（默认取 6mm）
    maximum node distance = 60mm, % 最大结点间距（默认取 60mm）
    border line width = \pgflinewidth, % 流程框边框宽度（默认取当前线条宽度）
    flow line width = \pgflinewidth, % 流程线线条宽度（默认取当前线条宽度）
    stealth length = 1.5mm,       % 箭头长度（默认取 1.5mm）
    stealth width = 1.0mm,        % 箭头宽度（默认取 1.0mm）
}
```

这些参数设置方式与常规的 TikZ 参数设置类似，即可以多个参数一起设置，也可以分开单独设置。`\flowchartset` 命令可以在导言区进行全局设置，也可以在需要的位置进行局部设置。

3 流程图绘制步骤

3.1 布置结点

使用类似“`\node [proc] (p1){表达式 1};`”TikZ 绘图命令,采用 `tikz-flowchart` 定义的“`proc`”、“`test`”、“`io`”或“`term`”的“`\node`”样式参数布置需要的流程框结点。

在布置结点时，如果前一个结点不是“`test`”样式，则可以使用“`join`”参数

自动与前一个结点建立连接, 如 “\node [proc, join] (p3) {表达式 3};”, 同时绘制对应流程线。

另外, 可以根据需要同时对布置的结点进行命名 (如 “p1”、“t1” 等), 以便后续对该结点进行引用。

3.2 布置坐标点

使用类似 “\node [coord, above = 0.5 of p1] (c1) {}; \cmark{1}” TikZ 绘图命令, 采用 tikz-flowchart 定义的 “coord” 的样式参数布置其它需要的坐标点 (用于流程线的转接)。同时, 也可以使用 “cmark” 命令为该点作出标记, 以方便调试流程线连线。该标记在使用 \usepackage[<debug>]{tikz-flowchart} 命令, 带 [<debug>] 可选参数引入宏包时, 将绘制这些标记点, 若引入宏包时无 [<debug>] 可选参数时, 则不绘制该标记点。

3.3 绘制流程线

首先使用类似 “\path(t1.south)--node[near start,right]{\$Y\$}(p2.north);” 的路径命令绘制流程线标注。

然后使用类似 “\draw[norm](t1.east)--(c6)|-(c4)--(c5);” 绘制命令绘制带有箭头的流程线。

在绘制流程线时, 可以使用 “lnorm”、“lfree” 或 “lcong” 指定颜色的样式绘制无箭头的流程线, 用 “norm”、“free” 或 “cong” 指定颜色的样式绘制有箭头的流程线。

建立先绘制南北方向流程线, 再绘制东西方向流程线。

在绘制流程线时, 可以使用 “dotnorm”、“dotfree” 和 “dotcong” 样式绘制流程线实心交点, 用 “cdotnorm”、“cdotfree” 或 “cdotcong” 样式绘制流程线空心交点。

对于不相交的流程线, 可以用 “connect” 样式进行绘制。

3.4 绘制其它图形

在使用`\node`、`\path`、`\draw`等命令时，除了可以使用`tikz-flowchart`宏包定义的样式外，也可以同时使用任何合法的TikZ绘图参数改变绘制属性。

同时，也可以使用任何合法的TikZ命令在流程图中绘制需要的图形元素。

4 流程图样例

4.1 改变流程线颜色

可以在导言区使用`\flowchartset`命令全局设置需要的绘图属性。

在进行参数设置时，可以仅指定需要的参数，如：

```
\usepackage{tikz-flowchart}
\flowchartset{
  free color = green, % 自由连线颜色（默认取 green）
  norm color = black, % 将常规连线颜色设置为 black
  cong color = red,   % 关联连线颜色（默认取 red）
}
.....
```

可以指定“norm”类型的颜色为“black”，可以得到图3所示的黑色流程线流程图。

4.2 改变流程框填充色

可以在导言区使用`\flowchartset`命令全局设置需要的绘图属性。

在进行参数设置时，可以仅指定需要的参数，如：

```
----- 局部设置绘图属性 -----
\flowchartset{
  proc fill color = orange!10,
  test fill color = green!30,
  io fill color = blue!30,
```

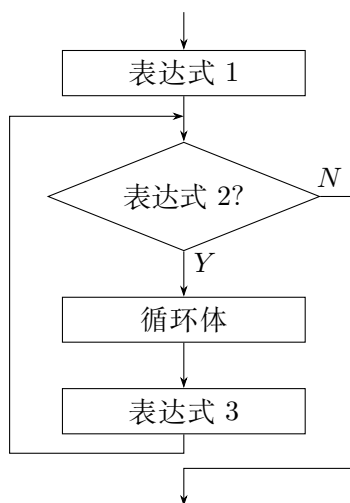



图 3: 黑色流程线流程图

```

term fill color = red!30,
proc text width = 6em,
test text width = 10em,
}
\begin{figure}[!htp]
\centering
\begin{tikzpicture}
\node [proc] (p1) {表达式 1};
\node [test, join] (t1) {表达式 2?};
\node [proc] (p2) {循环体};
\node [proc, join] (p3) {表达式 3};

\node [coord, above = 0.5 of p1] (c1) {};
\node [coord] (c2) at ($(p1)!0.35!(t1)$) {};
\node [coord, below = 0.25 of p3] (c3) {};
\node [coord, below = 0.2 of c3] (c4) {};
\node [coord, below = 0.5 of c4] (c5) {};
\node [coord, right = 0.5 of t1] (c6) {};
\node [coord, left = 0.5 of t1] (ct) {};
\node [coord] (c7) at (c3 -| ct) {};

\path (t1.south) -- node [near start, right] {$Y$} (p2.north);

```

```

\draw [norm] (t1.south) -- (p2.north);
\path (t1.east) -- node [near start, above] {$N$} (c6);
\draw [norm] (t1.east) -- (c6) |- (c4) -- (c5);

\draw [norm] (c1) -- (p1);
\draw [norm] (p3.south) |- (c7) |- (c2);
\end{tikzpicture}
\caption{黑色流程线流程图}
\label{fig:forloopblack}
\end{figure}

```

4.3 局部设置

也可以在需要的位置使用`\flowchartset{}` 命令进行需要的绘图局部设置。例如，可以使用如下代码绘制如图 4所示的彩色流程图。

```

% \flowchartset{
%   proc fill color = orange!10,
%   test fill color = green!30,
%   io fill color = blue!30,
%   term fill color = red!30,
%   proc text width = 6em,
%   test text width = 10em,
% }
%
% \begin{figure}[!htp]
%   \centering
%   \begin{tikzpicture}
%     % 布置结点单元
%     \node [term] (st) {开始};
%     \node [proc, join] (p1) {\verb|int divisor|};
%     \node [test, join] (t1) {\verb|n <= 1|};
%     \node [proc] (p2) {\verb|divisor = 2|};
%     \node [test, join] (t2) {\verb|divisor * divisor <= n|};
%     \node [test] (t3) {\verb|n % divisor == 0|};

```

```

% \node [proc] (p3) {\verb|divisor++|};
% \node [term, below = 1.6 of p3] (end) {结束};
% \node [proc, left = 4.8 of t2] (p4) {\verb|return 0|};
% \node [proc, right = 3.5 of p3] (p5) {\verb|return 0|};
% \node [proc, right = 5.8 of t3] (p6) {\verb|return 1|};
%
% 布置用于连接的坐标结点，同时为其布置调试标记点。
% \node [coord] (c1) at ($(p2.south)!0.5!(t2.north)$) {};
% \node [coord, below = 0.25 of p3] (c2) {};
% \node [coord, above = 0.5 of end] (c3) {};
% \node [coord, left = 0.5 of t2] (ct) {};
% \node [coord] (c4) at (c3 -| p5) {};
% \node [coord] (c5) at (c2 -| ct) {};
%
% 判断框连线，每次绘制时，先绘制一个带有一个固定
% 位置标注的路径(path)，然后再绘制箭头本身(arrow)。
% \path (t1.south) -- node [near start, right] {$N$} (p2.north);
% \draw [norm] (t1.south) -- (p2.north);
% \path (t1.west) -| node [near start, above] {$Y$} (p4.north);
% \draw [norm] (t1.west) -| (p4.north);
%
% \path (t2.south) -- node [near start, right] {$Y$} (t3.north);
% \draw [norm] (t2.south) -- (t3.north);
% \path (t2.east) -| node [near start, above] {$N$} (p6.north);
% \draw [norm] (t2.east) -| (p6.north);
%
% \path (t3.south) -- node [near start, right] {$N$} (p3.north);
% \draw [norm] (t3.south) -- (p3.north);
% \path (t3.east) -| node [near start, above] {$Y$} (p5.north);
% \draw [norm] (t3.east) -| (p5.north);
%
% 其它连线
% \draw [norm] (p3.south) |- (c5) |- (c1);
% \draw [norm] (p4.south) |- (c3);
% \draw [norm] (p4.south) |- (c3) -- (end);
% \draw [norm] (p5.south) -- (c4);

```

```

% \draw [norm](p6.south) |- (c3);
% \draw [norm](p6.south) |- (c3) -- (end);
% \end{tikzpicture}
% \caption{素数判定流程图}
% \label{fig:colorflowchart}
% \end{figure}
%

```

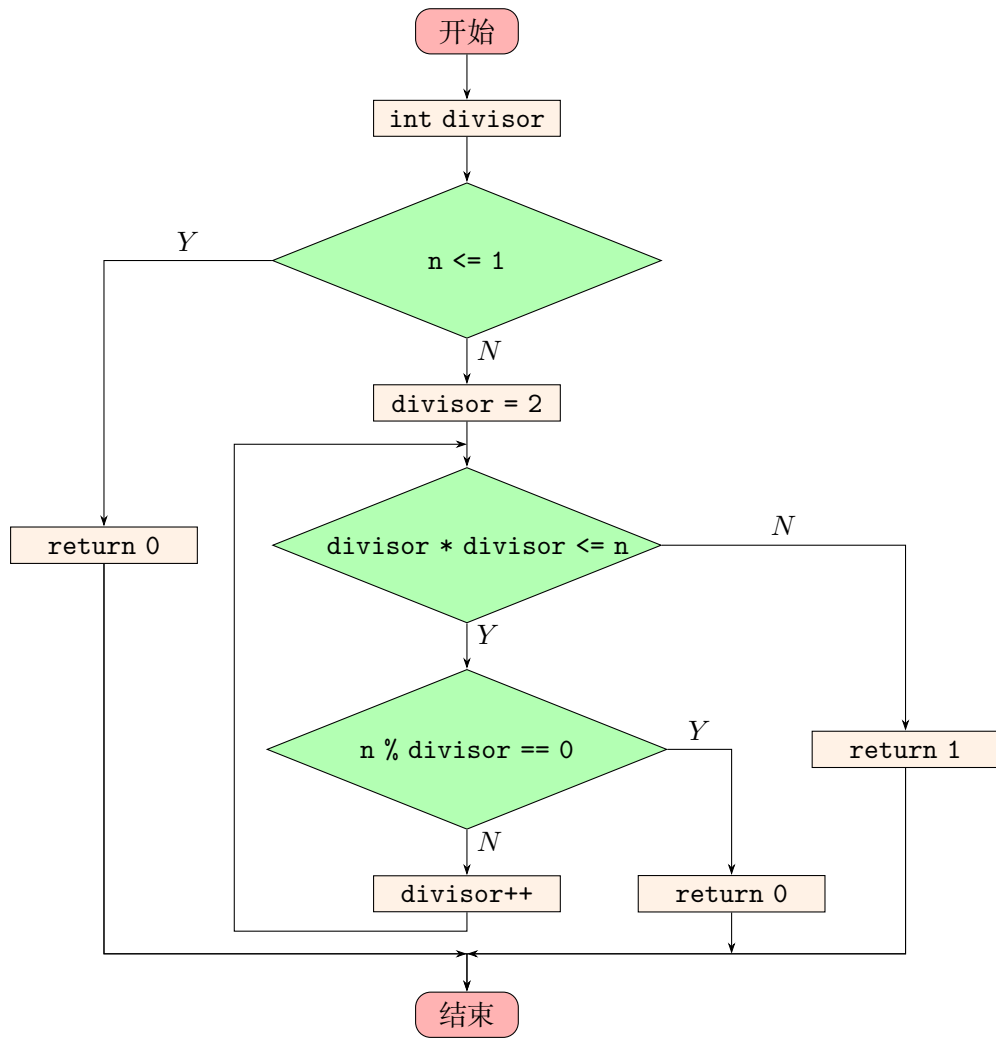


图 4: for 循环流程图

4.4 使用 TikZ 命令和参数

所有合法的 TikZ 命令和参数都可以应用于tikzpicture 绘图环境中。

5 代码实现

该宏包仅需要载入tikz 和xifthen 宏包，如果这些宏包没有载入，则自动载入这些宏包。

```
1 \RequirePackage{tikz}
2 \RequirePackage{xifthen}
3 %
```

5.1 宏包选项

使用 kvoptions 来处理传给该宏包的参数。

```
4 \RequirePackage{kvoptions}
5 \SetupKeyvalOptions{
6   family=flowchart,
7   prefix=flowchart@,
8   setkeys=\kvsetkeys
9 }
```

定义调试状态参数 (布尔类型)。

```
10 \DeclareBoolOption[false]{debug}
```

参数解析，获取定义的宏包参数。

```
11 \DeclareDefaultOption{}
12 \kvsetkeys{flowchart}{}
13 \ProcessKeyvalOptions*
```

载入需要的 TikZ 宏包运行库。

```
14 \usetikzlibrary{
15   arrows.meta,      % 箭头形状
16   shapes.geometric, % 几何形状
```

```

17 chains,          % 链式布局
18 calc,           % 坐标计算
19 }

```

5.2 配置命令

为`\flowchartset` 命令定义一个`pgfkeys` 族，所有配置参数（例如：`\normcolor`）等）都会存储在`/flowchartPGF` 键中。这能够确保这些配置参数不会覆盖别的同类参数。

```

20 \pgfkeys{
21   /flowchart/.is family,
22   /flowchart/.search also={/tikz},
23 }
24
25 \def\flowchartset{\pgfqkeys{/flowchart}}

```

然后，定义存储这些参数值的宏命令。

```

26 \flowchartset{
27   free color/.store in = \freecolor, % 自由连线颜色
28   norm color/.store in = \normcolor, % 常规连线颜色
29   cong color/.store in = \concolor, % 关联连线颜色
30   proc fill color/.store in = \procfillcolor, % 顺序处理框填充颜色
31   test fill color/.store in = \testfillcolor, % 判断框填充颜色
32   io fill color/.store in = \iofillcolor, % 输入/输出框填充颜色
33   term fill color/.store in = \termfillcolor, % 开始/结束框填充颜色
34   proc text width/.store in = \proctxtwd, % 顺序处理框宽度
35   test text width/.store in = \testtxtwd, % 判断框宽度
36   io text width/.store in = \iotxtwd, % 输入/输出框宽度
37   term text width/.store in = \termtxtwd, % 开始/结束宽度
38   chain direction/.store in = \chaindir, % 结点自动布置方向
39   minimum node distance/.store in = \minnodedis, % 最小结点间距
40   maximum node distance/.store in = \maxnodedis, % 最大结点间距
41   border line width/.store in = \bdlinewd, % 各类流程框边框宽度
42   flow line width/.store in = \flowlinewd, % 各类流程线线条宽度
43   stealth length/.store in = \stealthlen, % 箭头长度
44   stealth width/.store in = \stealthwd, % 箭头宽度
45 }

```

5.3 默认参数值

为各个参数设置默认值以确保预设的各个宏的值有效，这些值可以由用户单独进行修改，修改后的值会覆盖参数默认值。

```
46 \flowchartset{
47   free color = green, % 自由连线颜色 (默认取 green)
48   norm color = blue, % 常规连线颜色 (默认取 blue)
49   cong color = red, % 关联连线颜色 (默认取 red)
50   proc fill color = white, % 顺序处理框填充颜色 (默认取白色)
51   test fill color = white, % 判断框填充颜色 (默认取白色)
52   io fill color = white, % 输入/输出框填充颜色 (默认取白色)
53   term fill color = white, % 开始/结束框填充颜色 (默认取白色)
54   proc text width = 8em, % 顺序处理框宽度 (默认取 8em)
55   test text width = 5em, % 判断框宽度 (默认取 5em)
56   io text width = 6em, % 输入/输出框宽度 (默认取 6em)
57   term text width = 3em, % 开始/结束宽度 (默认取 3em)
58   chain direction = below, % 结点自动布置方向 (默认取 below)
59   minimum node distance = 6mm, % 最小结点间距 (默认取 6mm)
60   maximum node distance = 60mm, % 最大结点间距 (默认取 60mm)
61   border line width = \pgflinewidth, % 各类流程框边框宽度 (默认取当前线条
    宽度)
62   flow line width = \pgflinewidth, % 各类流程线线条宽度 (默认取当前线条宽
    度)
63   stealth length = 1.5mm, % 箭头长度 (默认取 1.5mm)
64   stealth width = 1.0mm, % 箭头宽度 (默认取 1.0mm)
65 }
```

5.4 样式定义

以下是所有绘制流程图中需要的样式定义。

```
66 \tikzset{
```

首先，定义结点布局方式：

```
67   start chain = going \chaindir, % 结点自动布置方向 (默认取 below)
68   node distance = \minnodedis and \maxnodedis, % 结点间距
69   every join/.style = {norm}, % 默认自动连接线的连线样式
```

其次，定义基础绘图样式：

```
70 % 流程框样式的基础样式
71 base/.style = {line width = \bdlinewidth, % 边框线宽
72               draw, % 绘制边框
73               on chain, % 沿布局方向绘制
74               on grid, % 沿网格布局
75               align=center, % 内容居中对齐
76               minimum height=2ex, % 流程框最小高度
77               },
```

接下来，定义proc、test、io、term 四个\textbackslash node 命令的绘图样式：

```
78 % 顺序处理框样式
79 proc/.style={base, % 基础样式
80             rectangle, % 矩形边框
81             text width=\proctxtwd, % 最大文本宽度（超过会自动换行）
82             fill=\procfillcolor, % 填充色
83             },
84 % 判断框样式
85 test/.style={base, % 基础样式
86             diamond, % 菱形边框
87             aspect=2.5, % 长高比例
88             text width=\testtxtwd, % 最大文本宽度（超过会自动换行）
89             fill=\testfillcolor, % 填充色
90             },
91 % 输入/输出框样式
92 io/.style={base, % 基础样式
93           trapezium, % 平行四边形
94           trapezium left angle=70, % 平行四边形左倾角
95           trapezium right angle=110, % 平行四边形右倾角
96           text width=\iotxtwd, % 最大文本宽度（超过会自动换
行）
97           fill=\iofillcolor, % 填充色
98           },
99 % 开始/结束框样式
100 term/.style={proc, % 基于 proc 样式
101             rounded corners=2.0mm, % 为矩形添加圆角属性
102             text width=\termtxtwd, % 最大文本宽度（超过会自动换行）
103             fill=\termfillcolor, % 填充色
104             },
```


再下来，定义流程线交点绘制样式：

```
105 % 流程连接点样式
106 connector/.style = {draw, % 绘制边框
107 circle, % 圆形
108 node distance=3cm, % 节点间距
109 },
110 % 绕接连线点样式（不相交的两个交汇路径）
111 connect/.style args={(#1) to (#2) over (#3) by #4}{
112 insert path={
113 let \p1=($(#1)-(#3)$), \n1={vecLen(\x1,\y1)},
114 \n2={atan2(\y1,\x1)}, \n3={abs(#4)}, \n4={#4>0 ?-180:180} in
115 (#1) -- ($(#1)!\n1-\n3!(#3)$) arc (\n2:\n2+\n4:\n3) -- (#2)
116 }
117 },
```

还需要定义流程线转角点node 命令样式：

```
118 % coord 结点样式（用于布置流程线连接点）
119 coord/.style={coordinate, % 笛卡尔坐标系
120 %on chain, % 沿布局方向绘制
121 %on grid, % 沿网格布局
122 node distance=6mm and 25mm, % 节点间距
123 },
```

为cmark 调试标记命令绘制样式：

```
124 % nmark 结点样式（用于布置调试坐标标记点）
125 nmark/.style={draw, % 绘制边框
126 cyan, % 青色
127 circle, % 圆形
128 font={\sffamily\bfseries}, % 字体
129 },
```

另外，需要定义各类流程线绘制样式：

```
130 % -----
131 % 无箭头连线样式
132 lnorm/.style={line width = \flowlinewidth, % 线宽
133 draw, % 绘制
134 \normcolor, % 颜色
135 },
136 lfree/.style={line width = \flowlinewidth,
```

```

137             draw,
138             \freecolor,
139         },
140     lcong/.style={line width = \flowlinewidth,
141         draw,
142         \congcolor,
143     },
144     % 流程线实心交点样式
145     dotnorm/.style={draw,           % 绘制
146         fill = \normcolor, % 填充颜色
147         \normcolor,         % 颜色
148     },
149     dotfree/.style={draw,
150         fill = \freecolor,
151         \freecolor,
152     },
153     dotcong/.style={draw,
154         fill = \congcolor,
155         \congcolor,
156     },
157     % 流程线空心交点样式
158     cdotnorm/.style={draw,         % 绘制
159         \normcolor, % 颜色
160     },
161     cdotfree/.style={draw,
162         \freecolor,
163     },
164     cdotcong/.style={draw,
165         \congcolor,
166     },
167     % 带箭头连线样式
168     norm/.style={line width = \flowlinewidth, % 线宽
169         --{Stealth[length=\stealthlen, % 箭头长度
170             width=\stealthwd, % 箭头宽度
171         ]
172     },
173     draw,           % 绘制
174     \normcolor,     % 颜色
175     },
176     free/.style={line width = \flowlinewidth,
177         --{Stealth[length=\stealthlen,
178             width=\stealthwd,

```

```

179         ]
180     },
181     draw,
182     \freecolor,
183 },
184 cong/.style={line width = \flowlinewidth,
185             -{Stealth[length=\stealthlen,
186                     width=\stealthwd,
187             ]
188     },
189     draw,
190     \congcolor,
191 },

```

最后，再定义一个流程线标注文本样式：

```

192 % 斜体字样式
193 it/.style={font={\small\itshape}},
194 }

```

5.5 调试命令定义

为便于绘制过程中，能够直观连接各个流程线转角点，定义 `cmark` 命令，以绘制转角点标记。

```

195 %% 判断是否为宏包传入了 debug 参数以打开调试功能，若没有传入 debug 参数，则
    关闭调试功能。
196 \ifflowchart@debug

```

传入了 `debug` 参数，创建用于调试的图层。

```

197 % 设置一个用于调试的标记符号图层，注意确保这一图层位于顶层
198 \pgfdeclarelayer{marx}
199 \pgfsetlayers{main,marx}

```

定义 `\textbackslash cmark` 命令。

```

200 \newcommand{\cmark}[2] [] {%
201 \begin{pgfonlayer}{marx}
202   \node [nmark] at (c#2#1) {#2};
203 \end{pgfonlayer}{marx}
204 }

```

未传入了 debug 参数，定义一个空的cmark 命令。

```
205 \else
206   \newcommand{\cmack}[2] [] {\relax}
207 \fi
```