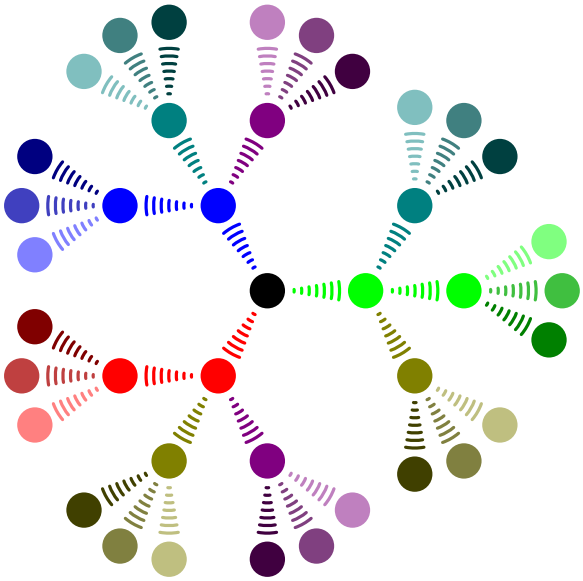
Ti*k*Z and PGF

Manual for version 1.10



```
\tikzstyle{level 1}=[sibling angle=120]
\tikzstyle{level 2}=[sibling angle=60]
\tikzstyle{level 3}=[sibling angle=30]
\tikzstyle{every node}=[fill]
\tikzstyle{edge from parent}=[snake=expanding waves,segment length=1mm,segment angle=10,draw]

\tikz [grow cyclic,shape=circle,very thick,level distance=13mm,cap=round]
  \node {} child [color=\A] foreach \A in {red,green,blue}
    { node {} child [color=\A!50!\B] foreach \B in {red,green,blue}
      { node {} child [color=\A!50!\B!50!\C] foreach \C in {black,gray,white}
        { node {} }
      }
    };
```

Für meinen Vater, damit er noch viele schöne TEX-Graphiken erschaffen kann.

TikZ 与PGF 宏包

1.10 版本使用手册

http://sourceforge.net/projects/pgf

Till Tantau
Institut für Theoretische Informatik
Universität zu Lübeck
tantau@users.sourceforge.net

中文翻译*李文军†
解放军炮兵学院
数学教研室
sydlee@gmail.com

2007年1月30日

目录

*时间比较紧,先发译了的前三篇
†错字或是错句请发电子邮件给我

1 介绍

PGF 宏包,这里“PGF”可以理解为“可移植图像格式(portable graphics format)” (如果你愿意的话... 可以为“漂亮的(pretty),好的(good),有用的(functional)”), 是一个使用内建形式的作图宏包.它定义了大量的用来绘图的TEX命令.例如, 代码`\tikz \draw (0pt,0pt) -- (20pt,6pt);`输出线 代码`\tikz \fill[orange] (1ex,1ex) circle (1ex);`输出.

在一定意义上来说,当你使用PGF,你是在编程绘你的图像,就像你使用TEX 是通过编程来实现你的文档一样.你会获得所有”TEX 式排版”的优点,如: 精确的位置,使用宏,当然还有超棒的打印效果.你同时也继承了所有的缺点,如: 陡峭的的学习曲线,非“所见即所得(WYSIWYG)”,微小的改动而需较长时间的重编译, 而且代码也不真的可以”表现”出你会画出什么样子的图形

1.1 系统的结构

PGF 包括一些不同的层

系统层: 这一层对“驱动”所做的事情进行了一个完全的抽象.这里的驱动是指一些程序如: `dvips` 或`dvipdfm` 它们可以把`.dvi`文件做为输入并且转化为一个`.ps`或是一个`.pdf`文件. (`pdftex`程序也常被计为一个驱动,尽管它并不把一个`.dvi`文件作为输入.不要太介意).每个驱动都有它自己的绘图的语法,这样就使人们为了得到可移植的图像而发愁.PGF 的系统层抽象出了这些不同.例如,系统命令`\pgfsys@lineto{10pt}{10pt}`把当前图`{pgfpicture}`的当前路径延伸到(10pt, 10pt). 系统命令会转化为不同的`\special`命令,关键看你在使用`dvips`, `dvipdfm`还是`pdftex`来处理这个文档. 系统层要做到尽可能的要求越低越好,因为每新增一条命令,就会增加把PGF 移入一个新驱动中要做的工作量.

作为一个用户,你可能不需要直接使用系统层.

基础层: 基础层为你提供了一系列的命令帮助你建立一个复杂的图形,相比于直接使用系统层来说,它要简单的多. 举个例子来说:系统层没有提供作圆的命令，是因为圆可以通过一些基本的贝塞儿曲线组合而成。但是，作为一个用户你希望一种简单的方法来创建一个圆（起码我是这样想的），而不是要用半页纸来写下贝塞儿曲线的坐标系方程. 因此基础层提供了一个`\pgfpathcircle`命令,它可以为你产生你所需曲线的坐标.

基础层是由一个*核心*构成的,而核心是由一些相互依赖的包组成,这些包只能以单块的形式被载入,同时这些包提供了更多特殊用途的命令(如节点管理,画交界面),因此也就扩展了核心的功能.例如,BEAMER宏包使用了核心,但并没有使用所有的基础层的附加包.

前端层: 一个前端(可能会有不只一个)是一系列命令或特殊语法的组合,它让我们更加简单的使用基础层.直接使用基础层所面临的一个问题就是这一层的命令往往写的太“详细”.举个例子,画一个简单的三角形,当使用基础层你可能甚至要使用5个命令: 第一条是在三角形的第一个端点外开始路径,第二条是延伸当前路径到第二个端点,第三条是延伸到第三个端点,第四条是封闭这条路径,第五点才是真正的画出这个三角形(相对的是填充它).当使用`tikz`前端的话,所有的这些都汇成一个简单的类METAFONT式命令:

```
\draw (0,0) -- (1,0) -- (1,1) -- cycle;
```

有几个不同类型的前端

- Ti*k*Z 前端是PGF 的默认前端.它让你可以使用PGF 的所有功能,却非常简单易学. 它的语法混合了METAFONT和PSTRICKS,当然也包括一些我自己的想法.这个前端*既不是*一个完全兼容METAFONT的前端层,*也不是*一个完全兼容PSTRICKS的前端层,当然也不打算兼而有之.

- pgfpict2e前端实现了标准LaTeX 的{picture}环境,它的命令如\line或\vector使用了PGF 基础层.这一层并非是必需的,因为pict2e.sty包在实现{picture}环境过程中已经做的足够好了.尽管如此,这个宏包的一些想法还是给了我们一个简单的示范,说明了一个前端是如何实现的.

通过映射从PSTRICKS 的命令到PGF 的命令来实现一个pgftricks前端也是可能做到的. 但是我并没有这样做,尽管这完全可以实现.因为一些本来在PSTRICKS 中可以工作的命令,可能不会正常工作, 特别是一些带有很强的PostScript技术的PSTRICKS 命令.

作为一个PGF 用户,你可能会使用前端层的命令也许还要配合一些基础层的命令.出于这个原因, 本手册将首先解释前端的使用,然后是基础层,最后是系统层.

1.2 与其它图形包的比较

PGF 的功能不仅仅局限于是一个TeX 的图形包.接下来,我试着对比PGF 系统与其它的宏包, 并给出一个理性的公平的比较

1. 标准的LaTeX的{picture}环境允许你建立一些简单的图形,但是比较少.这当然不是因为LaTeX 的设计者们缺少知识和想像力的原因.相反的,这正是{picture}环境为了更好的可移植性所付出的代价,它的可移植性表现在它可以与所有的后台驱动配合工作.

2. pstricks宏包已经足够的让你创建任何你想像到的图形,但是它完全不具有可移植性.更重要的是,它既不支持pdftex也不支持其它非PostScript代码之外的任何驱动.

   对比于PGF ,pstricks有更加广泛的支持基础.近十年来,用户贡献了大量针对各种问题的宏包,这些宏包都相当完善.

   Ti*k*Z 的语法比pstricks的语法更加的和谐一致,因为Ti*k*Z 是在"一种中央集权的方式"下开发的,而且开发者早已对pstricks的缺点了然于胸.

   请注意,一些纯操作上的技巧pstricks可以做的,对PGF 也许是一个不可能的任务.尤其是,pstricks可以使用强大的PostScript 语言,使之获得一些欺骗性的技巧比如内联函数图像描绘.

3. xypic是一个古老的作图宏包.但是,它既难用又难学,因为它的语法和文档显的有点晦涩.

4. dratex宏包是一个小的作图宏包.相比于其它宏包包括PGF ,它显得特别的小,这可以看成是它的优点,当然也可以看成是缺点.

5. metapost程序可以看成是PGF 的强大的替代品.但是,它是一个外部程序,这也是产生问题的根源. 一边要绘一个小的图形,同时还要编译它,这所花的时间要比使用PGF 来的多.可是,使用metapost所带来的最主要的问题是标签的包含.这在PGF 的使用中是非常简单就能做到的.

6. 对于那些不希望使用"编程"的方法来画图(这对Ti*k*Z 和其它以上所提到的宏包都是必需的)的用户来说,xfig程序是一个重要的替代品.已经有一个转化程序在开发过程中了,它可以把xfig图转化为Ti*k*Z 和PGF .

1.3 一个实用程序:页面管理

PGF 包包含一个特别的叫做pgfpages的子包,它可以用来把几个页面安排进一个页面中.这个包实际上跟绘图没有太大的关系, 但是它是PGF 的一部分,因为它大量的使用了PGF

pgfpages子包提供了一些命令可以安排几个"虚拟页面"到一个"物理页面".基本思路是这样的:当TeX 准备"输出"一个页面时, pgfpages中断这个输出,取而代之的是把这个要输出的页面放在一特别的盒子中.一旦,累积

到了足够多的“虚拟页面”,就把它们适当缩小并安排在一个”物理页面”中,然后*才真正的*输出.这种机制使你可以直接在LaTeX 内部创建“两页合一页”型的文档而需要使用任何其它的外部程序.

但是,pgfpages能做的不限于此.你可以用它来给你的页面打上logo和水印,可以把16个页面打印在一个页面上,可以给页面加上边框,等等.

1.4 如何阅读本手册

这个手册既解释了PGF 系统又解释了它的使用方法.内容的安排上粗略的根据“用户友好”方式来组织. 容易而经常使用的命令和子包将首先介绍,更低层而深奥的将后介绍.

如果你还没有安装PGF ,请先读如何安装.其次,阅读一下教程,对你来说也许是个好方法.最后,你也许希望要浏览一下Ti*k*Z 的使用说明.一般你不需要阅读关于基础层的章.如果你想要自己写一个前端或是想要把PGF 引入到一个新的驱动中,你只需要阅读关于系统层的部分.

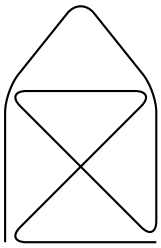
1.5 获得帮助

当你使用PGF 和Ti*k*Z 时需要帮助,请用以下方法:

1. 阅读这篇手册,至少你要阅读要你的问题可能所在的章节
2. 如果还没有解决问题,试着去sourceforge的PGF 和Ti*k*Z 开发页看看.(参见本文档的标题). 也许有人已经报告了近似的问题且成功被其他人所解决了.
3. 在网上你会找到很多的可获得帮助的论坛.在那里,你可以获得帮助,报告错误,加入邮件列表,等等.
4. 在报告一个错误,特别是有关安装的bug报告之前,请确认一下,是否真的是一个bug.尤其是要看一下执行TeX 过程中产生的.log文件.这个.log文件会显示出所有从正确的目录下被载入的正确文件.几乎所有的安装问题在看过.log文件后都可以被解决
5. *最后一招*你可以试着给我(作者)发email.我不介意收到email,只是我收的太多了.也正是因为如此, 我不保证会回复或是按时回复你的邮件.如果你发邮件到PGF 的邮件列表,你的问题得到成功解决的机会会大一些(自然的,只有当我有时间时才会读到这个邮件列表并回答你的问题)
6. 请不要打电话到我的办公室,如果你需要热线帮助,请去购买一个商业产品.

第一部分 I
教程与指南

为了能让你尽快的开始使用Ti*k*Z ,这篇手册以两个教程开始,而不是以长篇累牍安装配置说明开始. 这两个教程先解释了本系统所有基本的和一些略有进阶的功能特色,但并没有详细的展开.这一部分也包括了一些使用指南, 使用Ti*k*Z 创建图形时以什么样的步骤开始。



```
\tikz \draw[thick,rounded corners=8pt]
  (0,0) -- (0,2) -- (1,3.25) -- (2,2) -- (2,0) -- (0,2) -- (2,2) -- (0,0) -- (2,0);
```

2 教程:给Karl学生的一幅图

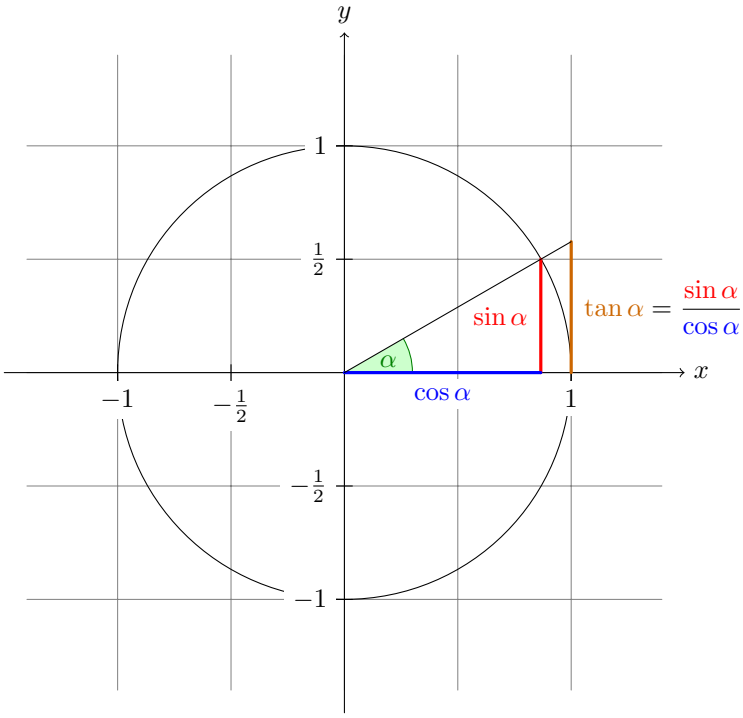
这是给PGF 和Ti*k*Z 新用户的一个教程.它并没有给出PGF 和Ti*k*Z 的全部功能说明, 只是给出了一些你马上就可以用的功能的说明.

Karl是一个高校的数学和化学老师.它经常使用LaTeX 的`{picture}`环境来建立一些学生问卷表或是考试卷.用它来作图的结果可以接受,但是过程却拖沓冗长.同时, 也会有一些问题,如直线的角度有点偏,圆显得不是那么的圆.自然的,他的学生们不会去关心这些直线角度是否精确,他们只是觉得Karl的考试太难了,而不会管那些直线画的有多漂亮. 但是Karl去对这个结果完成的不满意.

karl的儿子,更不满意这个结果(当然,他不用参加考试),他告诉karl 希望爸爸能尝试一个新的作图宏包. 稍有点迷惑的是,这个宏包有两个名字:第一个叫PGF.Karl下载下来并安装它.然后他发现在这个宏包里还有一个包叫做Ti*k*Z ,这个名字代表"Ti*k*Z ist *kein* Zeichenprogramm.[1]"Karl发现一个奇怪的事情,好像Ti*k*Z 的名字在说它并不能满足自己的要求.无论如何,他已经使用GNU 的软件很久了,"GNU not being Unix",试试看,也许还有转机.他的儿子提醒他, Ti*k*Z 的名字是在告诉人们, Ti*k*Z 不是一个使用鼠标或是手写板来画图的程序,相反的,它更像是一个画图语言.

2.1 问题陈述

Karl想要在下一次的学生问卷表中放一张图.他现在正在教学生们正弦和余弦函数.他想到得到的是这样一个图(想像中):



The angle α is 30° in the example ($\pi/6$ in radians). The sine of α, which is the height of the red line, is

$$\sin\alpha = 1/2.$$

By the Theorem of Pythagoras we have $\cos^2\alpha + \sin^2\alpha = 1$. Thus the length of the blue line, which is the cosine of α, must be

$$\cos\alpha = \sqrt{1 - 1/4} = \tfrac{1}{2}\sqrt{3}.$$

This shows that $\tan\alpha$, which is the height of the orange line, is

$$\tan\alpha = \frac{\sin\alpha}{\cos\alpha} = 1/\sqrt{3}.$$

2.2 设置环境

Ti*k*Z 中,要作一个图,在作图的开始的地方你要告诉TeX 或是LaTeX 你要开始作图了. 在LaTeX 中,使用环境`{tikzpicture}`,在plain TeX 中,你中需用`\tikzpicture`开始作图,用`\endtikzpicture` 结束作图.

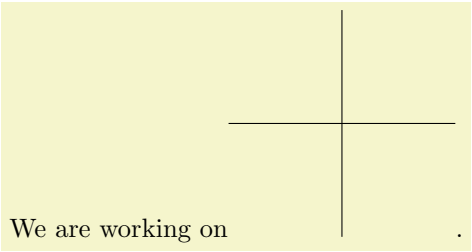
2.2.1 LaTeX 中设置环境

Karl,一个老LaTeX 用户,建立起了他的文件,如下:

[1] 德语,意思为Ti*k*Z 不是一个画图程序,是类似GNU的递归定义

```
\documentclass{article} % say
\usepackage{tikz}
\begin{document}
We are working on
\begin{tikzpicture}
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
\end{tikzpicture}.
\end{document}
```

运行它,可以通过执行pdflatex或是执行latex加dvips,运行的结果看上去是这个样子:




```
We are working on
\begin{tikzpicture}
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
\end{tikzpicture}.
```

诚然,没有得到整个图,但,至少我们已经建立了坐标轴.我们已经有了这些线,可以画坐标轴了. Karl突然有一种失落感,因为他觉得离他要的那张图差的还远着呢.

让我们仔细的看一下这些代码.首先,载入tikz.这个包相对于基础层来说,是一个所谓的"前端". 基础层,本手册也会有说明,但它有点..."基础",因此也很难学.这个前端通过提供一些简单的语法使事情变的简单多了.

在这个环境中,有二个\draw命令.它们的意思是:"此命令后,分号结束前,所指定的路径将会被画出" 第一条路径被指定为(-1.5,0) -- (0,1.5),意思是指"从点$(-1.5, 0)$到点$(0, 1.5)$的一条直线" 这里,点的位置被设定在一个特殊的坐标系中,此坐标系的单位长度初始值为 1cm.

Karl很高兴的发现这个环境自动的保留了足够的空间去环绕图形.

2.2.2 Plain TEX 中设置环境

Karl的妻子Gerda,也刚刚巧是一个数学老师,可她不是一个LATEX 用户,却是一个plain TEX 用户因为她偏好于用"老的方法".她也可能用Ti*k*Z.她应该使用\input tikz.tex, \tikzpicture,\endtikzpicture 而不是使用\usepackage{tikz},\begin{tikzpicture},\end{tikzpicture}.

因此,她这样使用:

```
%% Plain TeX file
\input tikz.tex
\baselineskip=12pt
\hsize=6.3truein
\vsize=8.7truein
We are working on
\tikzpicture
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
\endtikzpicture.
\bye
```

Gerda可以用pdftex或tex加dvips来排版这个文件.Ti*k*Z 将会自动辨别她使用了哪个驱动. 如果她要使用tex加dvipdfm,她需要做的要么是修改pgf.cfg文件,要么在载入tikz.tex 或pgf.tex之*前*的某处加上\def\pgfsysdriver{pgfsys-dvipdfm.def}

2.2.3 ConTEXt中设置环境

Karl的叔叔Hans使用ConTEXt.就像Gerda一样,Hans也可以使用Ti*k*Z. 他应该使用`\usemodule[tikz]`, `\starttikzpicture`, `\stoptikzpicture`, 而不是`\usepackage{tikz}`, `\begin{tikzpicture}`, `\end{tikzpicture}`

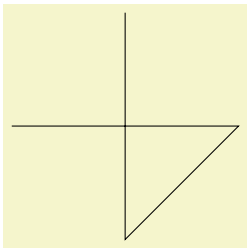
他的版本的例子类似如此:

```
%% ConTeXt file
\usemodule[tikz]

\starttext
  We are working on
  \starttikzpicture
    \draw (-1.5,0) -- (1.5,0);
    \draw (0,-1.5) -- (0,1.5);
  \stoptikzpicture.
\stoptext
```

Hans现在可以用`texexec`来排版他的文件了.

2.3 构建直线

在Ti*k*Z 在所有图形是最基本构成单元是路径.一条*路径*是指相连的一系列直线和曲线(这不包括全部的图形,但是让我们先略过复杂的部分).开始一条路径通过指定坐标系中的一个开始位置, 我们用一个圆括号包括中的点表示这个位置,如`(0,0)`.后面紧跟的是一系列的"路径扩充操作". 最简单最常用的是`--`.它后面要紧跟另一个坐标,它的作用是把路径用一条直线扩充到这个新的位置. 例如:如果我们想把上面坐标轴的两条路径合并成一条路径,下面的代码将会得到结果:



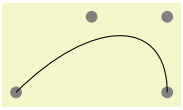
```
\tikz \draw (-1.5,0) -- (1.5,0) -- (0,-1.5) -- (0,1.5);
```

Karl有一点迷惑,因为这里没有使用`{tikzpicture}`环境,相反的,使用了一个`\tikz`命令. 这个命令要么只含有一个参数(放在一对花括号内,如:`\tikz{\draw (0,0) -- (1.5,0)}`,这将会得到_____) ,要么是搜索其后直到分号的所有内容, 并把它们放在一个`{tikzpicture}`环境中.作为一个首要的准则,所有的Ti*k*Z 画图命令必须作为`\tikz`的参数或者必须是在`{tikzpicture}`环境中.幸运的是,`\draw`命令只在这个环境中被定义过, 因此你碰巧犯错的概率是很小的.

2.4 构建曲线

Karl接下来想要做的是画一个圆,直线显然是做不出圆的.因此我们需要一些画曲线的方法. 为此,Ti*k*Z 提供了一个特别的语法,这里,我们需要一个或两个"控制点".其背景的数学方法并不平常,它的基本思想是:假设你在x点,并且第一个控制点是 y.那么这条曲线就会从x开始,"朝向 y的方向". 也就是说,曲线在x处的正切会指向 y.接下来,假设曲线的终点的z,并且第二个控制点是w.那么,实际上,曲线在点z处的正切就会通过w,最后结束于z点.

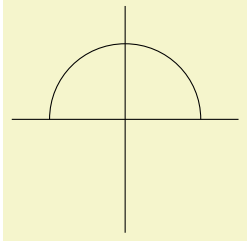
这里是一个例子(控制点已经被清晰的标注出来了)



```
\begin{tikzpicture}
  \filldraw [gray] (0,0) circle (2pt)
                   (1,1) circle (2pt)
                   (2,1) circle (2pt)
                   (2,0) circle (2pt);
  \draw (0,0) .. controls (1,1) and (2,1) .. (2,0);
\end{tikzpicture}
```

把路径以“曲线”的方式扩充的基本语法是:`.. controls` ⟨*first control point*⟩ `and` ⟨*second control point*⟩ `..` ⟨*end point*⟩.你可以省去`and` ⟨*second control point*⟩,此时第一个控制点将被使用两次.

现在,Karl可以给他的图加上第一个半圆了.



```
\begin{tikzpicture}
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (-1,0) .. controls (-1,0.555) and (-0.555,1) .. (0,1)
               .. controls (0.555,1) and (1,0.555) .. (1,0);
\end{tikzpicture}
```

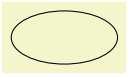
Karl对结果很满意,但他发现用这种方法指定一个圆实在是有点笨拙.幸运的是,我们还有其它的方法

2.5 构建圆型

为了画一个圆,可以使用构建路径操作`circle`.这个操作后面紧跟用圆括号括起来的半径值, 以下的例子说明了使用方法:(请注意,前面的点的位置是用来指定圆心的)

```
\tikz \draw (0,0) circle (10pt);
```

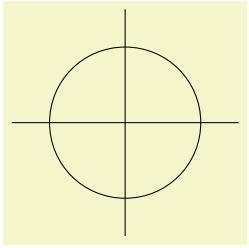
你也可以使用`ellipse`操作来把你的路径扩充成一个椭圆.与圆不同,你要指定两个半径,之间要用`and` 分开,其中一个代表半长轴,一个代表半短轴.



```
\tikz \draw (0,0) ellipse (20pt and 10pt);
```

`\draw (0,0) circle (1cm);` to draw the circle: 要画一个轴非水平和竖直,而是任意方向的椭圆(一个“旋转的椭圆”如:⬭),你要使用坐标变换,这将在以后解释. 顺便说一句,这个小椭圆的代码是:`\tikz \draw[rotate=30] (0,0) ellipse (6pt and 3pt);`.

那么,回到Karl的问题上,他可以写下:

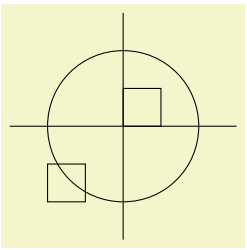


```
\begin{tikzpicture}
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
\end{tikzpicture}
```

此时,Karl已经注意到了这个圆比他想要的圆要小的多.他很高兴的了解到Ti*k*Z 有着强大的坐标变换操作,也可以轻易的放缩3倍.但是让我们先让它以现在的尺寸放在这吧,这样可以省点空间.

2.6 构建矩形

我们要做的下一件事就是要画出背景的网络线.有很多种方法可以产生这个效果. 例如,其中一种方法就是画很多个矩形.既然矩形这么的常用,也就需要专门给他们定义了一个语法表达式: 要在当前的路径中加入矩形,可以使用路径操作`rectangle`.这个操作之后要紧跟另一个坐标,这样就在当前的路径中加入了一个矩形,操作符前面的点和后面的点就是这个矩形对角线上的两个点.接下来, 让我们在这张图中加入两个矩形.



```
\begin{tikzpicture}
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \draw (0,0) rectangle (0.5,0.5);
  \draw (-0.5,-0.5) rectangle (-1,-1);
\end{tikzpicture}
```

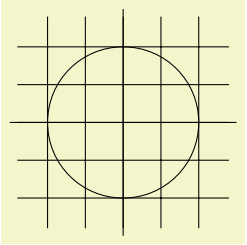
在一般情形下,这方法看上去很不错,但对于Karl的问题却用处不大.首先,我但需要一大堆这种矩形, 其次,他的图上还要有“未封闭”的边.

这时Karl知道了一个叫`grid`的路径操作,因此,他用了一个漂亮的`\draw`命令就简单的画出了四条竖线和四条横线.

2.7 构建网格线

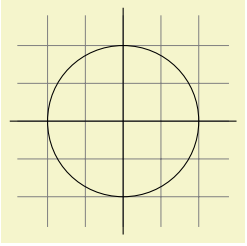
`grid`路径操作给,当前路径增加一个网格.它会在一个矩形内部画上一些线,形成一个网格. 这个矩形的一个顶点就是操作符前面的点,另一个顶点就是操作符后面的点.例如,代码`\tikz \draw[step=2pt] (0,0) grid (10pt,10pt);`生成  .注意一下`\draw`命令是如何使用一个可选参数来指定网格线的粗度的(还有`xstep`和`ystep`,用来分别指定步长).只要Karl学的够快,还有*很多* 类似的情形,它们都是使用这种可选参数来指定属性的.

Karl用了下面的代码.



```
\begin{tikzpicture}
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \draw[step=.5cm] (-1.4,-1.4) grid (1.4,1.4);
\end{tikzpicture}
```

再看一眼我们要作的那个图,Karl觉得如果网格线再淡一点就好了.(他的儿子告诉他,如果网格线不减淡的话, 它们会分散别人的注意).为了减淡网格线,Karl对`\draw`命令增加了两个选项来画这个网格线. 第一个,他用颜色`gray`来画网格.第二,他把线的粗度降低到`very thin`.最后,他交换了一下命令的使用顺序, 这样,网格线会先画上,其它的一切可以画在网格线上.



```
\begin{tikzpicture}
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
\end{tikzpicture}
```

2.8 加入一点风格

Karl也可以用`style=help lines`来代替选项`gray,very thin`. 风格,是一系列事先定义好的选项,它可以用来组织如何画一个图.当你说`style=help lines` 你是指“使用我(或是别人)已经设定好的风格来画辅助线”.只要Karl愿意,就可以画出那些网格线,也就是说使用颜色`blue!50`代替`gray`,(以后将会详细指出),他要用以下方法:

```
\tikzstyle help lines=[color=blue!50,very thin]
```

另一种代替的方法是:

```
\tikzstyle help lines+=[color=blue!50]
```

这样会增加一个[color=blue!50]选项.现在`help lines`的风格就包含了两个颜色选项了,后面一个会覆盖前面一个.

使用风格,可以让你的图形具有更多的变化.你可以以同样的方式简便的改变图的样子.

建立一个有颇有层次感的风格,你通过改造已有的某种风格来得到.所以,可以通过对已有的`grid`风格简单改造,就可以定义一个新的风格,叫做`Karl's grid`.Karl要这样写:

```
\tikzstyle Karl's grid=[style=help lines,color=blue!50]
...
\draw[style=Karl's grid] (0,0) grid (5,5);
```

你也可以把`style=`省略掉.这样的话,当Ti*k*Z 遇到一个它不知道的选项,它首先会检查这个选项是不是一个风格的名字.如果是,就使用这种风格.所以Karl也可以这样写:

```
\tikzstyle Karl's grid=[help lines,color=blue!50]
...
\draw[Karl's grid] (0,0) grid (5,5);
```

有些风格,比如`very thin`,清楚的表达出了它所指的只能是风格,你也无需再说`style=very thin`. 而其它的一些风格,像`help line`,对我来说,声明`style=help line`好像更加自然一些,没有别的原因,只是感觉好点.

2.9 作图选项

Karl还想知道有没有其它的选项可以影响路径.他已经看到`color=`⟨*color*⟩ 选项可以用来设定线的颜色.选项`draw=`⟨*color*⟩与之大概相同,只是它可以用来把线条的颜色和填充(Karl在填充那个角的扇形区域时,会使用到)的颜色设定的不同.

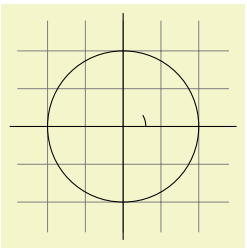
他发现使用风格`very thin`得到很细的线.Karl对此并不惊奇,他对以下用法也不感到丝毫陌生:`thin`风格得到细的线, `thick`风格得到粗的线,`very thick`风格得到很粗的线,`ultra thick`得到特别粗的线,还有`ultra thin`得到特别细的线(细到一些低分辨率的打印机或是显示器可能会无法正常显示). 但他却不晓得怎么样得到“一般”粗度的线.看上去`thin`是一个正确的选择.Karl觉得有点怪怪的,但是他的儿子解释说:LaTeX 中有两个命令分别叫做`\thinlines` 和`\thicklines`,其中`\thinlines` 给出的就是“一般”粗度的线,精确的来说,它的粗度就是字母“T”,“i”中竖线的粗度. 无论如何,Karl还想知道在`thin`和`thick`之间有没有“中间地带”.确实有一个,叫做`semithick`.

对于线条另一个有用的东西是可以把它们画成虚线或是点线,为此,可以使用`dashed`, 和`dotted`两种风格,分别得到_ _ 和.....两个选项都各包含疏松和紧密两种形式,分别叫做`loosely dashed`, `densely dashed`, `loosely dotted`, 和`densely dotted`.Karl如果确实非常非常需要的话,他也可以定义更复杂的虚线样式,使用`dash pattern` 选项,但是他的儿子却坚持认为:要非常谨慎的使用虚线,它会让人分散注意力.Karl的儿子声称:复杂的虚线样式是“魔鬼”,Karl的学生也不会去关注这些虚线样式.

2.10 构建弧形

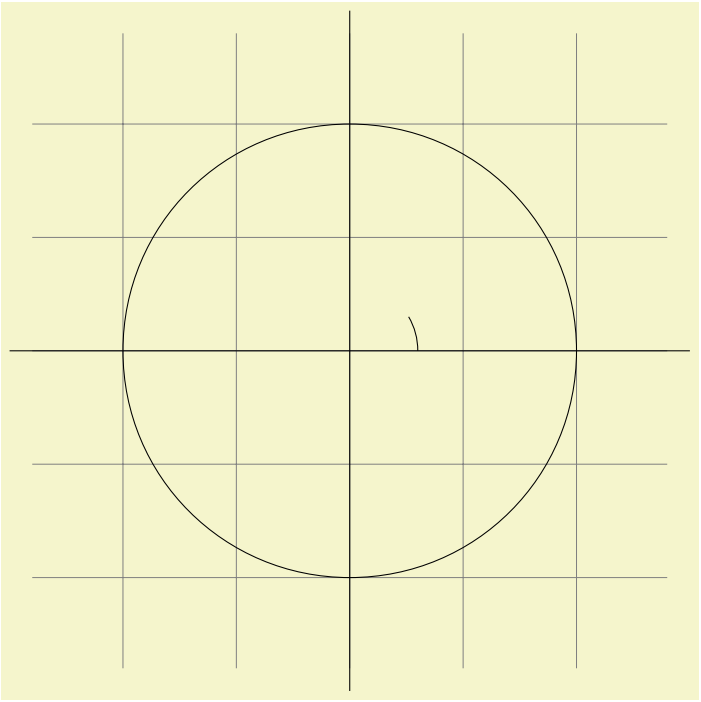
我们下面一个难点,是为这个角画出一道弧.为此,路径操作arc就派上用场了,它可以用来画圆或椭圆的一部分.arc操作符后面必须要跟有放在圆括号中的三个值,这三个值用冒号来分隔. 前二个值是角度,后一个值是半径.例如:(10:80:10pt)的意思是“从10度到80度,半径为10pt的一部分圆”.Karl显然只需要从0° 到30°的一段弧,半径应该相对小一点,大概三分之一圆的半径就可以了.这样得到(0:30:3mm).

当使用弧路径操作时,指定的弧会被放在当前指定的起始点之处,所以,让我们先“到达”起始点.



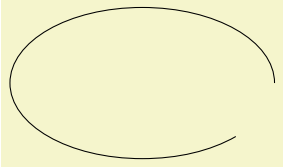
```
\begin{tikzpicture}
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \draw (3mm,0mm) arc (0:30:3mm);
\end{tikzpicture}
```

Karl觉得图真是的太小了,如果再不学会使用放缩的话,将很难继续下去了.因此,他增加了一个[scale=3]选项.他可以把这个选项加在每个\draw命令之后,但这样会很糟糕.于是, 他把选项加在了整个环境上,这样环境中的每处都会应用这个选项.



```
\begin{tikzpicture}[scale=3]
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \draw (3mm,0mm) arc (0:30:3mm);
\end{tikzpicture}
```

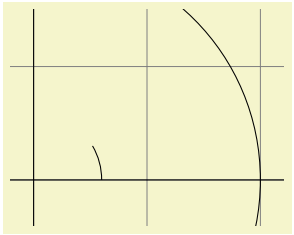
就像前面我们对圆的操作类似,你也可以指定“两个”半径来获得一个椭圆弧.



```
\tikz \draw (0,0) arc (0:315:1.75cm and 1cm);
```

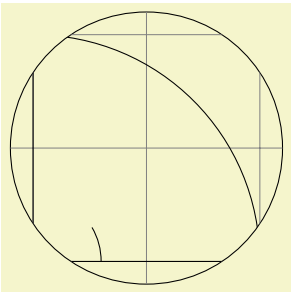
2.11 裁剪路径

为了在这篇手册中节省空间,我们最好对Karl的图进行一点裁剪,以让我们只集中精力于“感兴趣” 的部分.在Ti*k*Z 中,裁剪非常的方便.你可以使用`\clip`命令来裁剪接下来所有的绘图操作. 它就像`\draw`,但是它不画出任何东西,而只是使用给出的路径来裁剪接下来的所有图形.



```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \draw (3mm,0mm) arc (0:30:3mm);
\end{tikzpicture}
```

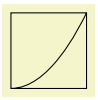
你可以同时作两件事:画路径*同时*裁剪它们.为此,可以使用`\draw`命令再加上`clip`选项.(这不会得到一个完整的图:你也可以使用`\clip`命令再加上`draw`选项.当然,这也不会得到一个完整的图:事实上,`\draw`只是`\path[draw]`的一个简写,`\clip`只是`\path[clip]` 的一个简写,当然你也可以使用`\path[draw,clip]`).这是一个例子.



```
\begin{tikzpicture}[scale=3]
  \clip[draw] (0.5,0.5) circle (.6cm);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \draw (3mm,0mm) arc (0:30:3mm);
\end{tikzpicture}
```

2.12 构建抛物线和正弦曲线

尽管Karl用不到,但他还是高兴的发现有三个路径操作`parabola` , `sin` 和`cos`可以分别画出抛物线,正弦和余弦曲线.对`parabola`路径操作来说,当前点必须位于抛物线上, 操作符之后紧跟的点也要在抛物线上.思考一下下面的例子.



```
\tikz \draw (0,0) rectangle (1,1)  (0,0) parabola (1,1);
```

也可以把顶点放在某处:

```
\tikz \draw[x=1pt,y=1pt] (0,0) parabola bend (4,16) (6,12);
```

`sin`和`cos`操作是在当前路径中加上区间$[0,\pi/2]$上的正弦和余弦曲线, 操作符前面的当前点就是曲线的起点,后面的点就是终点.这里有两个例子:

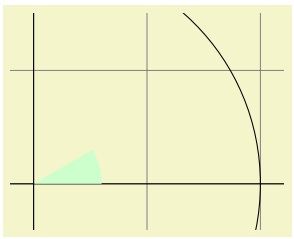
A sine ⌒ curve.

```
A sine \tikz \draw[x=1ex,y=1ex] (0,0) sin (1.57,1); curve.
```

```
\tikz \draw[x=1.57ex,y=1ex] (0,0) sin (1,1) cos (2,0) sin (3,-1) cos (4,0)
                            (0,1) cos (1,0) sin (2,-1) cos (3,0) sin (4,1);
```

2.13 填充和绘图

回到图上,Karl现在想要把那个角用淡绿色填充.因此,他没了用`\draw`命令,而是使用了`\fill` 命令,看看Karl做的:



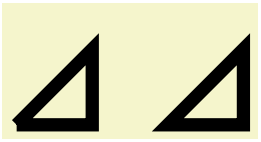
```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \fill[green!20!white] (0,0) -- (3mm,0mm) arc (0:30:3mm) -- (0,0);
\end{tikzpicture}
```

颜色`green!20!white`意思是20% 绿80% 白的混合.这种颜色表达式是允许的,因为PGF 使用的是Uwe Kern的`xcolor`宏包,有关颜色表达式的详情请参见此宏包的文档.

如果Karl不在路径的最后使用`--(0,0)`使得路径"封闭",那么会发生什么事呢? 在这种情况下,路径会自动的被封闭,所以它是可以省略的.事实上,以下面的方式写会更好些:

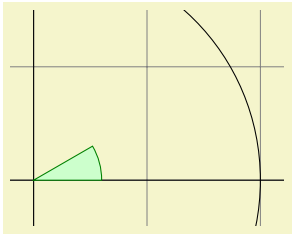
```
\fill[green!20!white] (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
```

`--cycle`会把第一个点和最后一点光滑的连接起来,使得当前的路径封闭起来(实际上是当前部分的当前路径)看下面的例子,体会一下其中的不同之处:



```
\begin{tikzpicture}[line width=5pt]
  \draw (0,0) -- (1,0) -- (1,1) -- (0,0);
  \draw (2,0) -- (3,0) -- (3,1) -- cycle;
  \useasboundingbox (0,1.5); % make bounding box higher
\end{tikzpicture}
```

你也可以通过使用`\filldraw`命令一次性的完成填充和画路径两件事. 这条命令首先画路径,然后填充它.这看上去用处不太大,但是你可以用它给填充部分和笔画部分指定不同的颜色.它们的参数说明如下:



```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \filldraw[fill=green!20!white, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

2.14 阴影

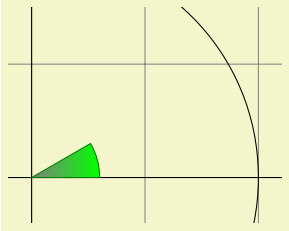
Karl在思考在没有可能可以通过*阴影*的方法让那个角变的更加精美呢,使用各种颜色使之有渐变效果,而不是只用一种颜色去填充它.为此,可以使用`\shade` 和`\shadedraw`, 前者用来作阴影,后者可以绘画的同时作上阴影.

```
\tikz \shade (0,0) rectangle (2,1)  (3,0.5) circle (.5cm);
```

默认的阴影是从灰到白的渐变.要指定不同的颜色,你可以使用选项:

```
\begin{tikzpicture}[rounded corners,ultra thick]
  \shade[top color=yellow,bottom color=black] (0,0) rectangle +(2,1);
  \shade[left color=yellow,right color=black] (3,0) rectangle +(2,1);
  \shadedraw[inner color=yellow,outer color=black,draw=yellow] (6,0) rectangle +(2,1);
  \shade[ball color=green] (9,.5) circle (.5cm);
\end{tikzpicture}
```

对Karl来说,下面会比较合适:



```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

可是,他聪明的认为,阴影通过会分散注意力而不会给图像带来任何好处.

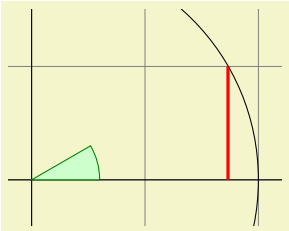
2.15 指定坐标

Karl现在要画上正弦和余弦曲线.他现在已经知道可以用`color=`选项来设定他要的线的颜色了. 可是有什么好的方法指定坐标吗?

有好几种方法来指定坐标.最简单的方法是这样说`(10pt,2cm)`,它的意思是x轴方向10pt, y轴方向2cm.你也可以省略单位,写成`(1,2)`,它的意思是"x轴向量的一倍,Y轴向量的两倍".这两个向量默认的长度分别为1cm.

为了指定极坐标中的点,使用`(30:1cm)`形式,它的意思是方向30度极径1cm.为了得到"圆周上$(\cos 30°, \sin 30°)$的点",这种形式显得特别有用.

你可以在一个坐标前面放一个`+`号或是两个,如`+(1cm,0cm)` , `++(0cm,2cm)`. 它们的意思是不同的:前者是指"从指定的位置向右移动1cm",后者是指"从当前的位置向上移动2cm, 并把此点作为新的指定点.".例如:我们可用如下方法画出正弦

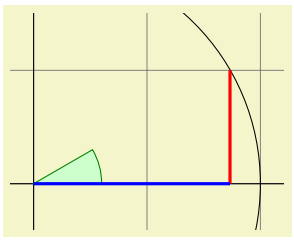


```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \filldraw[fill=green!20,draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
  \draw[red,very thick] (30:1cm) -- +(0,-0.5);
\end{tikzpicture}
```

Karl在这里用到了$\sin 30° = 1/2$这个知识点,但他非常担心他的学生们是否知道, 于是要找到一种好的方法来指定"从`(30:1cm)`点到x轴的垂足".事实上有一个特别的表达式, `(30:1cm |- 0,0)`,可以让Karl完成这件事.一般

的,(⟨p⟩ |- ⟨q⟩) 的意思是“过p点的一条竖直线与过q点的一条水平线的交点.

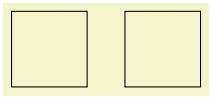
下面,我们开始作余弦线[2].一种方法是用(30:1cm |- 0,0) -- (0,0),另一种方法是在下面:我们从正弦线的结束开“继续”做下去.



```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \filldraw[fill=green!20,draw=green!50!black] (0,0) -- (3mm,0mm) arc
  (0:30:3mm) -- cycle;
  \draw[red,very thick]  (30:1cm) -- +(0,-0.5);
  \draw[blue,very thick] (30:1cm) ++(0,-0.5) -- (0,0);
\end{tikzpicture}
```

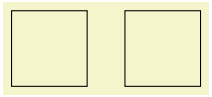
注意一下,在点(30:1cm)与点+(0,-0.5)之间没有--,详细的说,这条路径的意思可能解释为: “首先,点(30:1cm)告诉我要把笔移动到$(\cos 30°, 1/2)$.然后,又指定了另一个坐标,于是我把笔移动到那里,但中间并不画出任何东西.这个新的点是上一个点向下移动了半个单位,因此它的坐标是$(\cos 30°, 0)$.最后,我把笔移动到原点,但是这一次要画点东西(因为有操作符++)”.

再体会一下+与++的不同之处,看一下下面的例子:



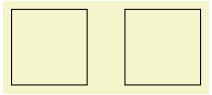
```
\begin{tikzpicture}
  \def\rectanglepath{-- ++(1cm,0cm)  -- ++(0cm,1cm)  -- ++(-1cm,0cm) -- cycle}
  \draw (0,0) \rectanglepath;
  \draw (1.5,0) \rectanglepath;
\end{tikzpicture}
```

通过对比,我们看出,当使用一个+号时,坐标是不同的.



```
\begin{tikzpicture}
  \def\rectanglepath{-- +(1cm,0cm)  -- +(1cm,1cm)  -- +(0cm,1cm) -- cycle}
  \draw (0,0) \rectanglepath;
  \draw (1.5,0) \rectanglepath;
\end{tikzpicture}
```

自然的,以上这此代码(不管是使用单个还是双个+)都可以写的更清晰更精简,像这样:



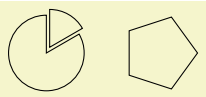
```
\tikz \draw (0,0) rectangle +(1,1)  (1.5,0) rectangle +(1,1);
```

Karl要做一个正切线$\tan\alpha$,使用极坐标和坐标变换的方法好像比较难于实现. 因此,他需要另一种指定坐标的方法:Karl可以指定线与线的交点作为某个坐标. 正切线从$(1,0)$点出发,它的终点正是两条线的交点,这两条线分别是:从$(1,0)$紧直向上的线和从原点出发经过(30:1cm)的线.表达式如下:

```
\draw[very thick,orange] (1,0) -- (intersection of 1,0--1,1 and 0,0--30:1cm);
```

接下来,最后给出两个例子,说明如何使用相对位置.注意这里的坐标变换选项(将在以后解释), 通常在需要用到移位时,它比相对位置用处更大.

[2]译者注:非余弦曲线

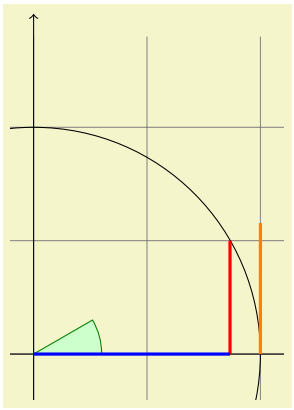


```
\begin{tikzpicture}[scale=0.5]
  \draw (0,0) -- (90:1cm) arc (90:360:1cm) arc (0:30:1cm) -- cycle;
  \draw (60:5pt) -- +(30:1cm) arc (30:90:1cm) -- cycle;

  \draw (3,0)  +(0:1cm) -- +(72:1cm) -- +(144:1cm) -- +(216:1cm) --
               +(288:1cm) -- cycle;
\end{tikzpicture}
```

2.16 增加箭头

Karl现在希望在坐标轴的顶点外加上箭头.他早就发现在一些图,甚至是某些科学杂志上的图, 也经常丢掉箭头,大概是因为他们的绘图程序做不到的原因吧.Karl觉得箭头应该在坐标轴的尽头,他的儿子也表示同意.但他的学生却不关心什么箭头.

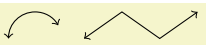
事实表明,加上箭头是非常的简便:karl给画坐标轴的命令都加上了->选项:



```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,1.51);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw[->] (-1.5,0) -- (1.5,0);
  \draw[->] (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \filldraw[fill=green!20,draw=green!50!black] (0,0) -- (3mm,0mm) arc
  (0:30:3mm) -- cycle;
  \draw[red,very thick]    (30:1cm) -- +(0,-0.5);
  \draw[blue,very thick]   (30:1cm) ++(0,-0.5) -- (0,0);
  \draw[orange,very thick] (1,0) -- (intersection of 1,0--1,1 and 0,0--30:1cm);
\end{tikzpicture}
```

如果Karl使用选项<- 而非->,箭头将会被放在路径的起始处. 而<->则会把路径的两端都加上箭头.

当然,可加上箭头的路径会有所限制,而非任何路径都可加上箭头.做为首要准则,你只能给一条“开放”的线加上箭头.例如,你不可以给一个矩形或是圆形加箭头.(你可以尝试一下,但我不能保证会发生什么事情,不管是在当前的版本,还是在以后的版本)但是,你可以给曲线或是分为多个区间的路径加箭头,如下面的例子所展示的:



```
\begin{tikzpicture}
  \draw [<->] (0,0) arc (180:30:10pt);
  \draw [<->] (1,0) -- (1.5cm,10pt) -- (2cm,0pt) -- (2.5cm,10pt);
\end{tikzpicture}
```

Karl仔细的看了一眼Ti*k*Z 画在坐标轴端点处的箭头,把它放大一点好像是这个样子⟶.样子看上去有点面熟,事实上,它就是TEX 的标准箭头,经常用在形如: $f\colon A \to B$.

Karl很喜欢这个箭头,特别是,它不像大部分宏包提供的箭头那样的略显粗壮. 但是,他确信,有的时候他也许会需要其它类型的一些箭头.为此,Karl可以说>=⟨*right arrow tip kind*⟩,这里⟨*right arrow tip kind*⟩用来指定箭头的类型.例如:如果Karl说>=stealth,那么他就是在告诉Ti*k*Z 他需要“stealth-fighter类型[3]”的箭头:

```
\begin{tikzpicture}[>=stealth]
  \draw [->] (0,0) arc (180:30:10pt);
  \draw [<<-,very thick] (1,0) -- (1.5cm,10pt) -- (2cm,0pt) -- (2.5cm,10pt);
\end{tikzpicture}
```

Karl正在疑惑这样一个带有军事味道名字的箭头是否真的必需,此时他的儿子告诉他, 微软的powerpoint中也使用了相同的名字,这使得Karl稍微的安定了下来.他同时做了个决定,要在某个时候让学生们来讨论一下这件事.

[3]类似战斗机,因而得名

除了stealth之外,还有很多其它预定义好的箭头可供Karl来挑选, 参见章节??.更进一步,他如果需要一个新的箭头的话,也可以自己定义.

2.17 辖域

Karl已经看到大量的可以控制如何产生路径的图形操作.常常,他需要把某一人选项加给所有的图形操作.例如,Karl也许想要画三条粗线,但是又不想双通常的方法去完成.

如果Karl要把某一个图形选项加给整个图,他只要简单的把选项参数传递给\tikz命令或是{tikzpicture}环境(Gerda要选项参数传递给\tikzpicture,Hans选项参数传递给\starttikzpicture).但是,如果Karl只想把图形选项应用到其中的一组命令而不是整个图,他就要把这些命令放在一个{scope}环境中,(Gerda使用\scope和\endscope, Hans使用\startscope和\stopscope).这个环境把这些图形选项作为自己的一个参数, 并把这些参数应用到其环境辖域之内的每个命令,而对环境辖域之外则没有任何影响.

这里有一个例子:

```
\begin{tikzpicture}[ultra thick]
  \draw (0,0) -- (0,1);
  \begin{scope}[thin]
    \draw (1,0) -- (1,1);
    \draw (2,0) -- (2,1);
  \end{scope}
  \draw (3,0) -- (3,1);
\end{tikzpicture}
```

辖域还有一个有趣的功效:对裁剪区域所做的任何改变都只在当前的一个辖域中. 这样,如果你在一个辖域的某处使用\clip命令,那么其结果是,\clip命令的作用范围直到辖域的结束.由于还没有其它的方法来"放大"裁剪区域,它有时就会派上用场了.

Karl早已明白,给一个命令(如\draw)加上参数,它只被应用到这个命令上.但有时情况看上去稍显复杂.首先,加在命令(如\draw)上的参数并非真的是命令的参数,它们实际上是"路径的参数",可以放在路径的任意处.因此,除了\draw[thin] (0,0) -- (1,0);, 你也可以这样写:\draw (0,0) [thin] -- (1,0);或\draw (0,0) -- (1,0) [thin]; 他们的效果都一样.这看上去有点奇怪,因为前面的例子中,好像应该是先画出从$(0,0)$到$(1,0)$线,"然后"thin 选项才开始作用于它.无论如何,大部分的图形选项仅仅作用于整个路径.事实上,如果你在一个路径上同时声名thin 和thick,那么后面一个选项将会"胜出".

当读到以上内容,Karl注意到仅仅"大部分"图形选项作用于整个路径.实际上,所有的坐标变换选项*并非*作用于整个路径,它们只作用于"路径上紧跟它们的". 我们这会儿将要详细的了解一下它.尽管如此,创建一条路径过程中给出的所有选项,都只适用于本条路径.

2.18 坐标变换

当你指定一个坐标如:(1cm,1cm),那么这个坐标的在页面上的什么地方?为了确定位置, Ti*k*Z,TEX,和PDF 或是PostScript都对给定的某一个坐标应用了一种坐标变换,以便可以确定点最终将放在页面的什么位置上.

Ti*k*Z 提供了众多的选项使你能够在PGF 的内部坐标系统中自由的进行坐标变换. 例如,xshift选项允许你把接下来所有的点移动一个数量.


```
\tikz \draw (0,0) -- (0,0.5) [xshift=2pt] (0,0) -- (0,0.5);
```

注意到,你可以"在路径中"更改坐标变换,这很重要,因为PDF 或PostScript还不支持这个功能. 原因是,PGF 只认识它自己的变换矩阵.

这里有一个稍微复杂点的例子:

```
\begin{tikzpicture}[even odd rule,rounded corners=2pt,x=10pt,y=10pt]
  \filldraw[fill=examplefill] (0,0)   rectangle (1,1)
    [xshift=5pt,yshift=5pt]   (0,0)   rectangle (1,1)
                  [rotate=30]   (-1,-1) rectangle (2,2);
\end{tikzpicture}
```

最有用的坐标变换是平移操作:`xshift`和`yshift`.`shift`功能是平移到一个指定的点, 形式是`shift={(1,0)}`或`shift={+(0,0)}`(括号是必需的,这样才能让TEX 不错把逗号当成是分隔符).`rotate`功能是用来旋转一个角度(还有一个`rotate around`, 功能是绕着某个指定的点旋转).`scale`功能是以一个倍数进行缩放.`xscale`,`yscale`只对x轴或y轴方向进行缩放(`xscale=-1`对应是的翻转).`xslant` 和`yslant`功能是倾斜. 如果这些和我这里还没有提到的所有坐标变换还不够你使用的话,`cm`选项将允许你使用任意的一个坐标变换矩阵.顺便说一句,Karl的学生,还没知道什么是变换矩阵.

2.19 重复: For循环

Karl的下一个目标是在坐标轴上加上一些小的标记点,分别在-1, $-1/2$, $1/2$, 和1.如果可以使用循环就更好了,尤其是你需要在每个不同的位置做同样的事情的时候.有各种各样的宏包可以做到.LATEX 也有类似的内部命令, `pstricks` 带有强大的`\mulitdo`命令.所有这些都可以与PGF 和Ti*k*Z 放在一起使用, 如果你很熟悉它们的话,自由的使用吧.但是PGF 还是引入了另一个命令,叫做`\foreach`, 我之所以引入它,是因为我已经记不得其它宏包的使用语法了.`\foreach`定义在`pgffor`宏包中, 你也可以独立于PGF 使用它.Ti*k*Z 自动的包含它.

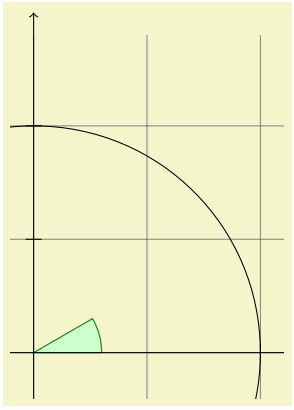
在一般形式下,`\foreach`使用很简单:

$x = 1$, $x = 2$, $x = 3$,

```
\foreach \x in {1,2,3} {$x =\x$, }
```

基本的语法表达式为`\foreach` ⟨*variable*⟩ `in {`⟨*list of values*⟩`}` ⟨*commands*⟩.在⟨*commands*⟩中,⟨*variable*⟩会被赋上不同的值. 如果⟨*commands*⟩没有以花括号开头,那么直到下一个分号之前的所有命令都会被当成⟨*commands*⟩.

Karl,要在数轴上标记点,他要如下这样做:



```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,1.51);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \filldraw[fill=green!20,draw=green!50!black] (0,0) -- (3mm,0mm) arc
  (0:30:3mm) -- cycle;
  \draw[->] (-1.5,0) -- (1.5,0);
  \draw[->] (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);

  \foreach \x in {-1cm,-0.5cm,1cm}
    \draw (\x,-1pt) -- (\x,1pt);
  \foreach \y in {-1cm,-0.5cm,0.5cm,1cm}
    \draw (-1pt,\y) -- (1pt,\y);
\end{tikzpicture}
```

事实上,有很多种方法建立标记点.例如:Karl可以把`\draw ...;`也放在花括号里. 他也可以这样使用:

```
\foreach \x in {-1,-0.5,1}
  \draw[xshift=\x cm] (0pt,-1pt) -- (0pt,1pt);
```

Karl很好奇一种更复杂的情形,例如有20个点要标记,该怎么办.看上去要细致的把所有的点都放`\foreach`需要的一个集合中,实在有点令人厌烦. 事实上,也可以在`\foreach`中使用`...`表达式,它可以用来迭代大量的数值(但是,这些值必须是无量纲的实数).用以下的例子说明:

```
\tikz \foreach \x in {1,...,10}
       \draw (\x,0) circle (0.4cm);
```

如果你在...之前放上了*两个*数,那么\foreach就会以它们之差为步长:


```
\tikz \foreach \x in {-1,-0.5,...,1}
      \draw (\x cm,-1pt) -- (\x cm,1pt);
```

我们可以嵌套循环再实现好玩的效果:

1,5	2,5	3,5	4,5	5,5
1,4	2,4	3,4	4,4	5,4
1,3	2,3	3,3	4,3	5,3
1,2	2,2	3,2	4,2	5,2
1,1	2,1	3,1	4,1	5,1

7,5	8,5	9,5	10,5	11,5	12,5
7,4	8,4	9,4	10,4	11,4	12,4
7,3	8,3	9,3	10,3	11,3	12,3
7,2	8,2	9,2	10,2	11,2	12,2
7,1	8,1	9,1	10,1	11,1	12,1

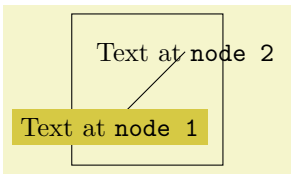
```
\begin{tikzpicture}
  \foreach \x in {1,2,...,5,7,8,...,12}
    \foreach \y in {1,...,5}
    {
      \draw (\x,\y) +(-.5,-.5) rectangle ++(.5,.5);
      \draw (\x,\y) node{\x,\y};
    }
\end{tikzpicture}
```

\foreach还可以有更加技巧性的使用,但一切都还得你的思想了.

2.20 添加文字

至此,Karl对这个图形已经非常的满意了.但是,最重要的部分,还没有文字标签.

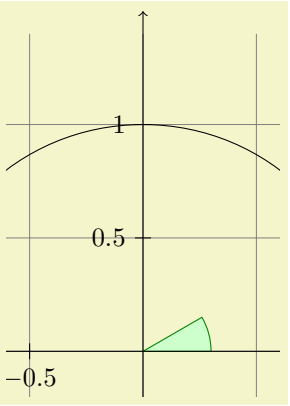
Ti*k*Z 提供了一个十分简单而有强大的系统去帮你在图形的指定位置处添加文字及其外框, 后者略显复杂.基本的思想如下:当Ti*k*Z 在创建路径的过程中,一旦它在路径中的某处遇到关键字node,它就会先去读*节点说明*.关键字node后面常常跟有一些选项,然后是一些放在花括号中的文字.这些文字被放在一个普通的TEX 盒子中(如果节点说明直接跟在一个坐标后面,实际上也经常是这样的,那么Ti*k*Z 通过施展一点魔力甚至可以把"逐字打印文本"[4]放在这些盒子中)并把它们放在当前位置处, 即最后指定的位置(根据所给的选项,有可能会漂移一点).但是,只有当路径完全被画出/ 填充/阴影/裁剪/别的什么,之后,所有的节点才会被画上.




```
\begin{tikzpicture}
  \draw (0,0) rectangle (2,2);
  \draw (0.5,0.5) node [fill=examplefill]
                      {Text at \verb!node 1!}
    -- (1.5,1.5) node {Text at \verb!node 2!};
\end{tikzpicture}
```

[4] verbatim text

显然,Karl不想仅仅把节点“放在”指定的位置上,而是希望放在这个位置的左边或右边. 为此,每个需放入图像中的节点都被“装备”上几个*锚点*.例如：`north`锚点位于外形上边缘的中间，`south`位于底部，`north east`锚点位于右上角.当你给出选项`anchor=north` ，那么这么北锚点就会被放在当前位置，因而节点说明中的文字被放在当前位置的下方. Karl用这种方法作出他的标记点：

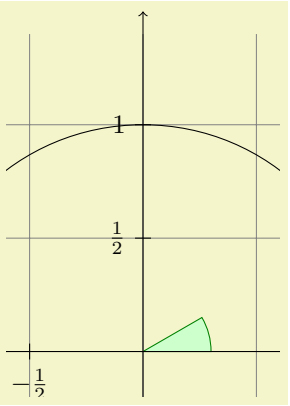



```
\begin{tikzpicture}[scale=3]
  \clip (-0.6,-0.2) rectangle (0.6,1.51);
  \draw[step=.5cm,style=help lines] (-1.4,-1.4) grid (1.4,1.4);
  \filldraw[fill=green!20,draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
  \draw[->] (-1.5,0) -- (1.5,0);   \draw[->] (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);

  \foreach \x in {-1,-0.5,1}
    \draw (\x cm,1pt) -- (\x cm,-1pt) node[anchor=north] {$\x$};
  \foreach \y in {-1,-0.5,0.5,1}
    \draw (1pt,\y cm) -- (-1pt,\y cm) node[anchor=east] {$\y$};
\end{tikzpicture}
```

很上去已经够美观了.Karl现在可以通过使用这些锚点来放置更多的文字元素了. 但是，Karl在想，要把节点放在一个点的*下方*,却要用一个北锚点,这有点跟直觉不符. 出于这个原因,还可以使用一个叫`below`的选项,他的作用与`anchor=north`一样. 类似的,`above right`与`anchor=south east`一样.另外,`below` 有一个可选的尺寸参数. 如果给出,那么外框将会额外的向下平移参数指定的距离.因此,`below=1pt`可用来把文字标签放在一个点的下方,并额外的向下平移1pt.

Karl对这些标记点不是十分满意.他希望看到1/2 或是$\frac{1}{2}$而不是0.5,原因有二, 其一是可以保持于TEX 和Ti*k*Z 的完美一致,其二是有些位置,如1/3 或π理所当然的要选用“数学化”的标记,而不是“数学化”的标记.另一方面,他的学生喜欢0.5胜过1/2 ,因为他们通常都不喜欢分数.

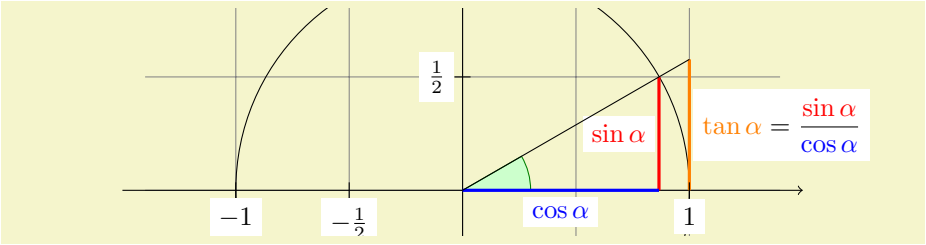
Karl现在面临一个问题`\foreach`语句中,位置`\x`仍然还要写成`0.5`,因为Ti*k*Z 不会知道`\frac{1}{2}`指的是什么位置.另一方面,排版出来的文字确是需要写成`\frac{1}{2}`. 为了解决这个问题,`\foreach`给出了一个特别的表达式:不再使用一个变量`\x`了,Karl现在可以指定两个(或是更多)的变量,这些变量要用斜线分隔开来,如`\x / \xtext`.然后,`\foreach`迭代集合中的元素也要写成⟨*first*⟩/⟨*second*⟩的形式.在每次迭代中,`\x`将被赋值为⟨*first*⟩, `\xtext`将被赋值为⟨*second*⟩.如果⟨*second*⟩没有给出,那么⟨*first*⟩将会再次被使用. 这样,关于标记的新代码如下:




```
\begin{tikzpicture}[scale=3]
  \clip (-0.6,-0.2) rectangle (0.6,1.51);
  \draw[step=.5cm,style=help lines] (-1.4,-1.4) grid (1.4,1.4);
  \filldraw[fill=green!20,draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
  \draw[->] (-1.5,0) -- (1.5,0); \draw[->] (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);

  \foreach \x/\xtext in {-1, -0.5/-\frac{1}{2}, 1}
    \draw (\x cm,1pt) -- (\x cm,-1pt) node[anchor=north] {$\xtext$};
  \foreach \y/\ytext in {-1, -0.5/-\frac{1}{2}, 0.5/\frac{1}{2}, 1}
    \draw (1pt,\y cm) -- (-1pt,\y cm) node[anchor=east] {$\ytext$};
\end{tikzpicture}
```

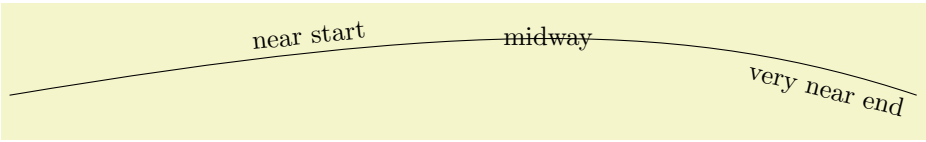
Karl对这个结果非常满意了,但是他的儿子指出这还没达到完美:后面的网格线和圆干扰了这几个数字,降低了它们的可读性.Karl对此关不太关心(他的学生们根本不会注意到),但是他的儿子坚持声称有一种简易的解决方法:Karl需要增加一个`[fill=white]`选项,用白色来填充文本框的背景.

Karl下面要做的事就是要添加标签如:$\sin\alpha$.为此,他希望把标签就在“线的中间” 要达到这个目的,不能再向往常一样把标签`node {$\sin\alpha$}`直接放在路径的最后面(这样会把图上的文字标签放在线的末端),而是要把它直接加在`--`与坐标的之间.默认的情形下,这样就会把标签放在线的中间,但是也可以`pos=`选项来调整它. 同样的,像选项`near start` 和`near end`也可以用来调整标签的位置.




```
\begin{tikzpicture}[scale=3]
  \clip (-2,-0.2) rectangle (2,0.8);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \filldraw[fill=green!20,draw=green!50!black] (0,0) -- (3mm,0mm) arc
  (0:30:3mm) -- cycle;
  \draw[->] (-1.5,0) -- (1.5,0) coordinate (x axis);
  \draw[->] (0,-1.5) -- (0,1.5) coordinate (y axis);
  \draw (0,0) circle (1cm);

  \draw[very thick,red]
    (30:1cm) -- node[left=1pt,fill=white] {$\sin \alpha$} (30:1cm |- x axis);
  \draw[very thick,blue]
    (30:1cm |- x axis) -- node[below=2pt,fill=white] {$\cos \alpha$} (0,0);
  \draw[very thick,orange] (1,0) -- node [right=1pt,fill=white]
    {$\displaystyle \tan \alpha \color{black}=
      \frac{{\color{red}\sin \alpha}}{\color{blue}\cos \alpha}$}
    (intersection of 0,0--30:1cm and 1,0--1,1) coordinate (t);

  \draw (0,0) -- (t);

  \foreach \x/\xtext in {-1, -0.5/-\frac{1}{2}, 1}
    \draw (\x cm,1pt) -- (\x cm,-1pt) node[anchor=north,fill=white] {$\xtext$};
  \foreach \y/\ytext in {-1, -0.5/-\frac{1}{2}, 0.5/\frac{1}{2}, 1}
    \draw (1pt,\y cm) -- (-1pt,\y cm) node[anchor=east,fill=white] {$\ytext$};
\end{tikzpicture}
```

你也可以把标签放在曲线上,并且通过添加选项`sloped`让他们旋转以让他们与线的倾斜度相吻合. 这里有一个例子.




```
\begin{tikzpicture}
  \draw (0,0) .. controls (6,1) and (9,1) ..
    node[near start,sloped,above] {near start}
    node {midway}
    node[very near end,sloped,below] {very near end} (12,0);
\end{tikzpicture}
```

最后只剩下要给图形加上文字说明这件事了.主要的困难是要把很长的一段文字标签放在一个宽度有限的地方上,因此合适的断行是必需的.幸运的是,Karl可以使用选项`text width=6cm` 来得到渴望的效果.最后,完整的代

码如下:

```
\begin{tikzpicture}[scale=3,cap=round]
  % Local definitions
  \def\costhirty{0.8660256}

  % Colors
  \colorlet{anglecolor}{green!50!black}
  \colorlet{sincolor}{red}
  \colorlet{tancolor}{orange!80!black}
  \colorlet{coscolor}{blue}

  % Styles
  \tikzstyle{axes}=[]
  \tikzstyle{important line}=[very thick]
  \tikzstyle{information text}=[rounded corners,fill=red!10,inner sep=1ex]

  % The graphic
  \draw[style=help lines,step=0.5cm] (-1.4,-1.4) grid (1.4,1.4);

  \draw (0,0) circle (1cm);

  \begin{scope}[style=axes]
    \draw[->] (-1.5,0) -- (1.5,0) node[right] {$x$} coordinate(x axis);
    \draw[->] (0,-1.5) -- (0,1.5) node[above] {$y$} coordinate(y axis);

    \foreach \x/\xtext in {-1, -.5/-\frac{1}{2}, 1}
      \draw[xshift=\x cm] (0pt,1pt) -- (0pt,-1pt) node[below,fill=white] {$\xtext$};

    \foreach \y/\ytext in {-1, -.5/-\frac{1}{2}, .5/\frac{1}{2}, 1}
      \draw[yshift=\y cm] (1pt,0pt) -- (-1pt,0pt) node[left,fill=white] {$\ytext$};
  \end{scope}

  \filldraw[fill=green!20,draw=anglecolor] (0,0) -- (3mm,0pt) arc(0:30:3mm);
  \draw (15:2mm) node[anglecolor] {$\alpha$};

  \draw[style=important line,sincolor]
    (30:1cm) -- node[left=1pt,fill=white] {$\sin \alpha$} (30:1cm |- x axis);

  \draw[style=important line,coscolor]
    (30:1cm |- x axis) -- node[below=2pt,fill=white] {$\cos \alpha$} (0,0);

  \draw[style=important line,tancolor] (1,0) -- node[right=1pt,fill=white] {
    $\displaystyle \tan \alpha \color{black}=
    \frac{{\color{sincolor}\sin \alpha}}{\color{coscolor}\cos \alpha}$}
    (intersection of 0,0--30:1cm and 1,0--1,1) coordinate (t);

  \draw (0,0) -- (t);

  \draw[xshift=1.85cm]
    node[right,text width=6cm,style=information text]
    {
      The {\color{anglecolor} angle $\alpha$} is $30^\circ$ in the
      example ($\pi/6$ in radians). The {\color{sincolor}sine of
        $\alpha$}, which is the height of the red line, is
      \[
      {\color{sincolor} \sin \alpha} = 1/2.
      \]
      By the Theorem of Pythagoras ...
    };
\end{tikzpicture}
```

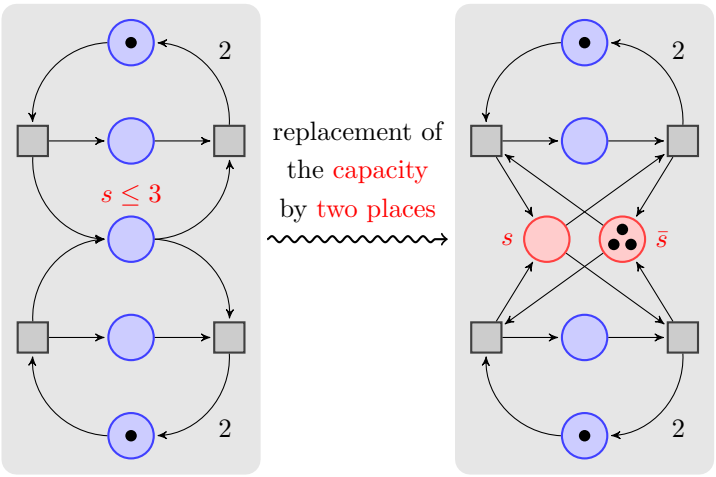
3 教程:Hagen的一个Pretri net图

在第二个教程中,我们将要探索一下Ti*k*Z 和PGF 的结点机制.

Hagen明天必须要做一个关于Pretri net[5]网演讲,这是他最喜欢分布系统形式. Hagen过去大多是用黑板来来完成这件事,而且别人通常也不会反对. 可是这次不行了,一方面是因为他的听众们最近被一些精美的幻灯片宠坏了,另一方面他的同事也希望他能用一个绘图程序来制作这个Petri nets.他们学院的一个教授向他推荐Ti*k*Z ,Hagen决定要试试.

3.1 问题表述

在演讲中,Hagen要创建一个示意图来展示一个有位置容积[6]的网络是如何被一个不含容积的网络来模拟的.想像中,图形应该是这样:



3.2 建立环境

要建这个图,Hagen要像前面教程里的Karl一样,先载入Ti*k*Z .但这次,他还需要载入一些其它的*库包*,这些是Karl不需要的.这些库包包含一些额外的定义,比如说一般情况下用不到的一些箭头,当你要使用它们时一定要明确的声明载入这些包.Hagen需要载入三个库:箭头库,用来在图形中使用特殊的箭头;“蛇型库”,用来在路径中间使用蛇型线;背景库,用来描绘图的两个主要部分的两个矩形背景.

3.2.1 LaTeX 中建立环境

在LaTeX 中,使用:

```
\documentclass{article} % say

\usepackage{tikz}
\usetikzlibrary{arrows,snakes,backgrounds}

\begin{document}
\begin{tikzpicture}
  \draw (0,0) -- (1,1);
\end{tikzpicture}
\end{document}
```

[5] “Petri Nets” 源自1962年Carl A. Petri的博士论文.类似网络图形,又具有数学理论的建模工具

[6] Pretri net的术语,可以不必理会

3.2.2 Plain TEX 中建立环境

在plain TEX中,使用:

```
%% Plain TeX file
\input tikz.tex
\usetikzlibrary{arrows,snakes,backgrounds}
\baselineskip=12pt
\hsize=6.3truein
\vsize=8.7truein
\tikzpicture
  \draw (0,0) -- (1,1);
\endtikzpicture
\bye
```

3.2.3 ConTEXt 中建立环境

在ConTEX 中,使用:

```
%% ConTeXt file
\usemodule[tikz]
\usetikzlibrary[arrows,snakes,backgrounds]

\starttext
  \starttikzpicture
    \draw (0,0) -- (1,1);
  \stoptikzpicture
\startext
```

3.3 结点简介

原则上,我们已经知道该如何创建Hagen所要的图了(除了可能会用到的蛇形线,我们以后会说到);我们先画一个浅灰色的大矩形,然后加上大量的圆和小矩形,最后加上一些箭头.

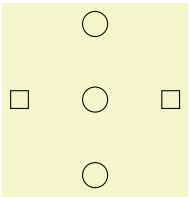
但是,这种方法有很多的弊端：首先，我们以后如果对图作更改将会变的很难.例如,如果我们要在Petri nets中加上更多的位置（这些圆在Petri nets网中被叫做位置）,那么所有的坐标都需要重新计算一遍.第二,这个Petri nets图的原代码将不那么易读,你看到的只是一堆又臭又长的坐标和命令的混在一起,而看不到Petri net的深层次的结构.

幸运的是,Ti*k*Z 提供了一种强大的机制,可以避免这种事情的发生,那就是结点.我们已经学习过了结点的使用,在前面我们用结点给Karl的图加上标签.在这个教程里将会看到结点还具有更为强大的功能.

一个结点就是一个图的一小部分.要创立一个结点,你只要给出一个位置,在那里结点及其外框将会一起被创建.一个带有`circle`型外框的结点将被画成一个圆,一个带有`rectangle`型外框的结点将被画成一个矩形,等等.一个结点也可以包含一些文字,Karl曾经用他来显示文字.最后,一个结点可以给它一个名字以便后面可以引用它.

在Hagen的图中,我们要使用结点来创立Petri net图的位置和过渡[7](圆是位置,矩形是过渡).我们先开始做Petri net图左侧的上半幅图.在这个上半幅图中有三个位置和两个过渡.用不着画三个圆两个矩形,我们用三个带有`circle`外框的结点和两个带有`rectangle`外框的结点.

[7]也是术语



```
\begin{tikzpicture}
  \path ( 0,2) node [shape=circle,draw] {}
        ( 0,1) node [shape=circle,draw] {}
        ( 0,0) node [shape=circle,draw] {}
        ( 1,1) node [shape=rectangle,draw] {}
        (-1,1) node [shape=rectangle,draw] {};
\end{tikzpicture}
```

Hagen发现这不太象最后的图,但是也算是一个良好的起步了.

我们仔细的看一下代码.整个图只包含一条路径.如果忽略node操作的话,好像对这条路径什么也没有做:这里只有一个几个坐标排在这里,中间什么也没有.实际上,就算对这条路径动了点手脚,比如说是画条直线到某处或是画条曲线到某处, \path命令对最终的路径也不会做任何操作.因此，所有的魔法肯定都在node命令中.

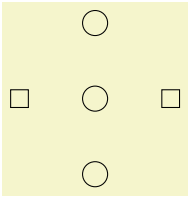
在前面的教程中我们知道,一个node命令将会在当前坐标处增加一些文本.这样,五个结点就被放置在不同的位置上了.在上面的代码中,没有加入任何文字(因为用了一个空括号{}).但为什么我们还是看到东西了?答案是node操作中的draw选项:它画出了一个“文本的外框”.

因此,代码(0,2) node [shape=circle,draw] {}的意思就是:“在主路径中,移动到点(0,2).然后,暂时中止对主路径的描绘而先创建结点.这个结点的样子是一个circle围绕着一个空的文本.这个圆先被draw出,但是没有被填充或是别作它用.一旦整个结点完成了,它会被保存下来直到主路径完成,然后才会被画出”然后接下来(0,1) node [shape=circle,draw] {},它的作用是:“移动到点(0,1)继续描绘主路径,然后也在这个位置处创建一个结点.这个结点是一样,只有在主路径完成后才会被显示.”如此这般下去.

3.4 使用At表达式放置结点

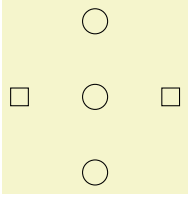
Hagen现在知道了node操作是如何在路径中增加结点的,但是看上去用\path操作来创建路径显得的不够聪明,使用了大量多余的移位操作,而这一切仅仅是为了放置结点.他很高兴的了解到还有一些更加理性的放置结点的方法.

首先,node操作允许使用如下形式at (⟨*coordinate*⟩),它可以直接把结点放在需要的地方,而回避了前面提到的规则,即结点要放在它紧跟着的坐标处.Hagen现在可以写下:



```
\begin{tikzpicture}
  \path node at ( 0,2) [shape=circle,draw] {}
        node at ( 0,1) [shape=circle,draw] {}
        node at ( 0,0) [shape=circle,draw] {}
        node at ( 1,1) [shape=rectangle,draw] {}
        node at (-1,1) [shape=rectangle,draw] {};
\end{tikzpicture}
```

现在Hagen得到的还是一条空路径,但是至少这条路径不再含有一些奇怪的移位操作了.看上去这里还有改善的余地:\node命令是\path node命令的简写形式,因此Hagen可以写成:

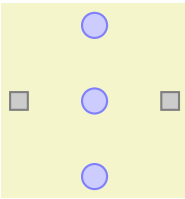


```
\begin{tikzpicture}
  \node at ( 0,2) [circle,draw] {};
  \node at ( 0,1) [circle,draw] {};
  \node at ( 0,0) [circle,draw] {};
  \node at ( 1,1) [rectangle,draw] {};
  \node at (-1,1) [rectangle,draw] {};
\end{tikzpicture}
```

比较于前一种表达式来说,Hagen更喜欢这种.注意到Hagen省略了shape=,这是因为Ti*k*Z 允许你在不至于混淆的情况下,省略shape=,同样的情形还适用于color=.

3.5 使用风格

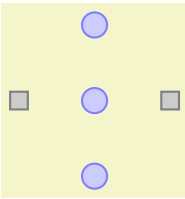
感觉到有点新奇,Hagen尝试着让结点更漂亮一点.在最后的图中,圆和矩形都要被不同的颜色填充上，代码为：



```
\begin{tikzpicture}[thick]
  \node at ( 0,2) [circle,draw=blue!50,fill=blue!20] {};
  \node at ( 0,1) [circle,draw=blue!50,fill=blue!20] {};
  \node at ( 0,0) [circle,draw=blue!50,fill=blue!20] {};
  \node at ( 1,1) [rectangle,draw=black!50,fill=black!20] {};
  \node at (-1,1) [rectangle,draw=black!50,fill=black!20] {};
\end{tikzpicture}
```

这样图形变的漂亮了，但同时代码变丑了。理想中，我们希望我们的代码能传递这样的信息“这里有三个位置二个过渡‘’，而不是用太多的命令来让这里或那里要填充颜色。

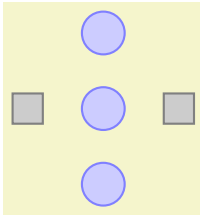
为了解决这个难题，Hagen使用了风格。他分别给位置和过渡定义了不同的风格：



```
\tikzstyle{place}=[circle,draw=blue!50,fill=blue!20,thick]
\tikzstyle{transition}=[rectangle,draw=black!50,fill=black!20,thick]
\begin{tikzpicture}
  \node at ( 0,2) [place] {};
  \node at ( 0,1) [place] {};
  \node at ( 0,0) [place] {};
  \node at ( 1,1) [transition] {};
  \node at (-1,1) [transition] {};
\end{tikzpicture}
```

3.6 结点大小

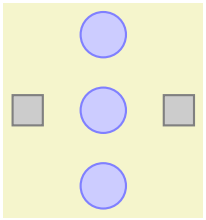
在Hagen开始命名和连接这些结点这前，让我们先确定一下结点是否都已是它们最终的样子了。不是，它们太小了。事实上，Hagen想不通它们为什么还会有这么一个尺寸，毕竟里面包含的文本是空的啊。原因是Ti*k*Z自动在文本周围增加一些空间.这个空间的大小可以用`inner sep`选项来给定.因此,为了增加结点的尺寸,Hagen可以这样写:



```
\tikzstyle{place}=[circle,draw=blue!50,fill=blue!20,thick]
\tikzstyle{transition}=[rectangle,draw=black!50,fill=black!20,thick]
\begin{tikzpicture}[inner sep=2mm]
  \node at ( 0,2) [place] {};
  \node at ( 0,1) [place] {};
  \node at ( 0,0) [place] {};
  \node at ( 1,1) [transition] {};
  \node at (-1,1) [transition] {};
\end{tikzpicture}
```

但是,这确实不是得到我们想要效果的最好方法.更好的替代方法是使用`minimum size`选项.这个选项允许Hagen指定这个结点的最小尺寸.如果这个结点确实需要变的更大以容纳更长的文本,它还可以变大,但是如果文本是空的,那么它的大小就是`minimum size`.这个选项也要以用来确保一系列包含不同文本内容的结点可以有一个统一的大小尺寸. 选项`minimum height` 和`minimum width`允许你单独的指定结点的最小高度和最小宽度.

因此,Hagen需要做的是给结点提供一个`minimum size`.安全起见,他同时设定了`inner sep=0pt`.这就保证了结点将确实处于`minimum size`而不是另外一种情况,即:你指定了一个很小的最小尺寸,而这个最小尺寸还要包含系统自动增加的空间尺寸.



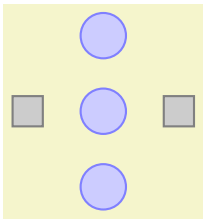
```
\tikzstyle{place}=[circle,draw=blue!50,fill=blue!20,thick,
                   inner sep=0pt,minimum size=6mm]
\tikzstyle{transition}=[rectangle,draw=black!50,fill=black!20,thick,
                        inner sep=0pt,minimum size=4mm]
\begin{tikzpicture}
  \node at ( 0,2) [place] {};
  \node at ( 0,1) [place] {};
  \node at ( 0,0) [place] {};
  \node at ( 1,1) [transition] {};
  \node at (-1,1) [transition] {};
\end{tikzpicture}
```

3.7 命名结点

Hagen的下一个目标是用箭头把结点连接起来.这看上去是一件棘手的事情,因为这些箭头不应该是从结点的中间出来的,而是从它的边缘的某处,另外,Hagen也非常的不愿意手工去计算这些点的位置.

幸运的是,PGF 将会帮计算好这一切.但是,首先他要给这些结点命名以便我们在后面可以引用它们.

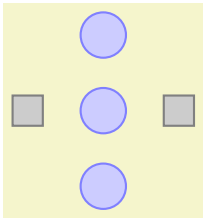
有两种方法来命名一个结点.第一种是用`name=`选项.第二种方法是`node`操作符后的括弧里写上你希望的名字.Hagen觉得第二种方法有点奇怪,但是他将很快改变他的想法.



```
% ... setup styles
\begin{tikzpicture}
  \node (waiting 1)  at ( 0,2)     [place] {};
  \node (critical 1) at ( 0,1)     [place] {};
  \node (semaphore)  at ( 0,0)     [place] {};
  \node (leave critical) at ( 1,1) [transition] {};
  \node (enter critical) at (-1,1) [transition] {};
\end{tikzpicture}
```

Hagen高兴的发现这些名字能帮助更好的理解代码.结点的名字是任意的,但是不应该包括逗号,句号,括弧,冒号和一些其它的特殊字符. 但是,可以包括下划线和连字符.

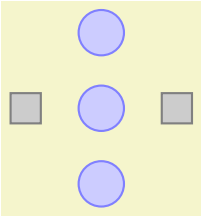
结点操作的语法表达式非常的灵活,命名结点,`at`定位,其它的一些选项,这些都没有固定的顺序.事实上你甚至可以在`node`和放置文本的花括号之间放多个选项块,让他们堆积在一起.你也可以把它们任意的重排,也许下面的比较不错:



```
\begin{tikzpicture}
  \node[place]      (waiting 1)      at ( 0,2) {};
  \node[place]      (critical 1)     at ( 0,1) {};
  \node[place]      (semaphore)      at ( 0,0) {};
  \node[transition] (leave critical) at ( 1,1) {};
  \node[transition] (enter critical) at (-1,1) {};
\end{tikzpicture}
```

3.8 使用相对位置放置结点

尽管Hagen仍然希望把结点连上,但是他首先希望解决另外一个问题:结点的放置.尽管他偏爱`at`表达式,但在这个特例中,他希望以某个结点“相对于其它结点”的相对位置来放置它们.因此,Hagen希望,不管`waiting 1`结点在什么位置,`critical 1`结点始终应该在`waiting 1`结点的下方.有不同的方法可以做到,但是在Hagen的实例中最好的办法是使用`below of`选项.



```
\begin{tikzpicture}
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical] {};
  \node[transition] (leave critical) [right of=critical] {};
  \node[transition] (enter critical) [left of=critical]  {};
\end{tikzpicture}
```

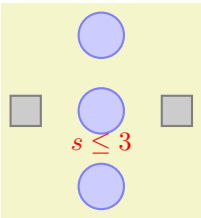
`below of`与其它类似的选项是这样来确定位置的:它把结点放在你指定的某个方向上距离为`node distance`远的地方.`node distance`是指两个结点中心的距离而不是边缘的距离.

尽管以上的代码与原先的产生一样的效果,但是Hagen可以把这段代码给他的同事们看,他们无需看到图就知道大概是什么样子的了.

3.9 给结点加上标签

在看到Hagen如何连接这些结点之前，让我们先在结点的下面加上相应的容积“$s \le 3$”.为此,有两种方法:

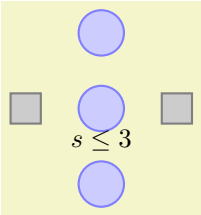
1. Hagen可以在`semaphore`结点的`north`锚点上方增加一个新的结点.




```
\begin{tikzpicture}
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical] {};
  \node[transition] (leave critical) [right of=critical] {};
  \node[transition] (enter critical) [left of=critical]  {};

  \node [red,above] at (semaphore.north) {$s\le 3$};
\end{tikzpicture}
```

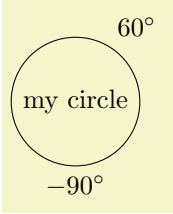
这是一个常用的方法,而且它“永远可行”

2. Hagen可以使用特别的`label`选项.这是一个结点的选项,它可以使得另一个结点加在带有此选项结点的边上. 想法是这样的:当我们建立`semaphore`结点时, 我们希望能够表现出另一个带有容积的结点在它的上方.为此,我们使用选项`label=above:$s\le 3$`.这个选项可以解释为:我们想要一结点位于`semaphore`结点的上方,这个结点应该为“$s \le 3$.” 除了`above`我们还可以使用`below left`等等,它们应放在冒号或是一个数字(如60)的前面.



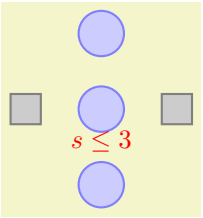

```
\begin{tikzpicture}
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical,
                                      label=above:$s\le3$] {};
  \node[transition] (leave critical) [right of=critical] {};
  \node[transition] (enter critical) [left of=critical]  {};
\end{tikzpicture}
```

也可以给出多个`label`选项,这将会画出多个标签.




```
\tikz
  \node [circle,draw,label=60:$60^\circ$,label=below:$-90^\circ$] {my circle};
```

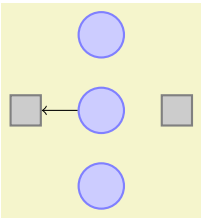
Hagen对这个label选项不完全满意,因为这个标签并不是红色的.为了达到这个目的,有两个选择:第一,他可以重新定义every label的风格.第二,他可以给标签结点增加选项.这些选项要紧跟在label=之后,因此写为label=[red]above:$s\le3$. 但是,这有是并不奏效,因为TEX 会认为]结束了整个semaphore结点的操作序列.所以,Hagen不得不增加一个花括号写成label={[red]above:$s\le3$}.这看上去有点丑,所以Hagen决定重新定义every label的风格.




```
\begin{tikzpicture}
  \tikzstyle{every label}=[red]
  \node[place]      (waiting)                              {};
  \node[place]      (critical)       [below of=waiting]    {};
  \node[place]      (semaphore)      [below of=critical,
                                      label=above:$s\le3$] {};
  \node[transition] (leave critical) [right of=critical]   {};
  \node[transition] (enter critical) [left of=critical]    {};
\end{tikzpicture}
```

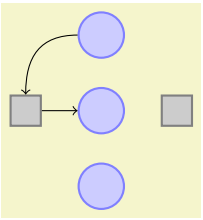
3.10 连接结点

现在是时候该连接结点了.让我们先从简单的开始,即从enter critical到critical的一条直线. 我们希望这条线从enter critical的右侧开始,结束于critical的左侧.为此,我们可以使用结点的*锚点*. 每个结点都定义了位于其边缘或是其内的一系列的锚点.例如,center锚点位于结点的中心,west锚点位于结点的左边,等等. 要获得一个结点的坐标,我们可以这样来引用:结点名跟一个点,再加上锚点的名称:



```
\begin{tikzpicture}
  \node[place]      (waiting)                              {};
  \node[place]      (critical)       [below of=waiting]    {};
  \node[place]      (semaphore)      [below of=critical]   {};
  \node[transition] (leave critical) [right of=critical]   {};
  \node[transition] (enter critical) [left of=critical]    {};
  \draw [->] (critical.west) -- (enter critical.east);
\end{tikzpicture}
```

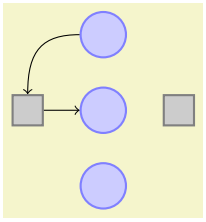
下面让我们来处理从waiting 到enter critical的曲线.我们要使用曲线和controls来实现它.



```
\begin{tikzpicture}
  \node[place]      (waiting)                              {};
  \node[place]      (critical)       [below of=waiting]    {};
  \node[place]      (semaphore)      [below of=critical]   {};
  \node[transition] (leave critical) [right of=critical]   {};
  \node[transition] (enter critical) [left of=critical]    {};
  \draw [->] (enter critical.east) -- (critical.west);
  \draw [->] (waiting.west) .. controls +(left:5mm) and +(up:5mm)
                            .. (enter critical.north);
\end{tikzpicture}
```

Hagen明白了如何添加所有的边了,但是整个过程看上去有点笨拙而缺乏变通.而且,代码让人更加难以理解图形的结构而不是更加的清晰易懂.

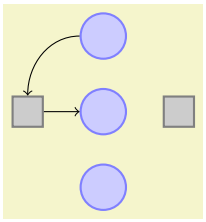
因此,让我们为这些边来改进代码.首先,Hagen可以不用锚点.



```
\begin{tikzpicture}
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical] {};
  \node[transition] (leave critical) [right of=critical] {};
  \node[transition] (enter critical) [left of=critical]  {};
  \draw [->] (enter critical) -- (critical);
  \draw [->] (waiting) .. controls +(left:8mm) and +(up:8mm)
                      .. (enter critical);
\end{tikzpicture}
```

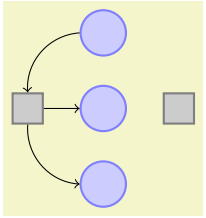
Hagen对于此代码能够正确显示觉得有点诧异.毕竟,Ti*k*Z 是怎么知道从`enter critical` 到`critical`的线实际上应该是从它们的边界开始的呢.一旦当Ti*k*Z 遇到一个结点名作为一个“坐标” 时,它将会“聪明的”选择这个结点的适当的锚点.Ti*k*Z 会根据你下一步的行动选择结点边缘上的一个锚点,连接到下一个坐标或是控制点.精确的规则有点复杂,但是它选择的点几乎总是正确的—-就算不正确,Hagen还是有办法可以手工指定的.

Hagen可以把曲线操作再简化一点.可以通过使用一个特殊的路径操作来达到目的:`to`操作.这个操作带有很多选项(你甚至可以自己定义一些新的选项).有一对选项Hagen可能会用得上:`in` 和`out`.这些选项用来指定角度,一条曲线以这个角度离开或是达到出发点或是目标点坐标.若没有这些选项,则会画出一条直线:



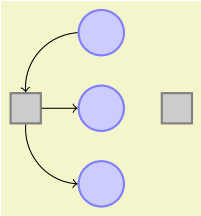
```
\begin{tikzpicture}
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical] {};
  \node[transition] (leave critical) [right of=critical] {};
  \node[transition] (enter critical) [left of=critical]  {};
  \draw [->] (enter critical) to                 (critical);
  \draw [->] (waiting)        to [out=180,in=90] (enter critical);
\end{tikzpicture}
```

`to`操作还带有一个选项,它甚至更适合于Hagen的问题:`bend right`选项.这个选项也是指定角度,但是这个角度仅仅指定了曲线向右侧弯曲的角度:



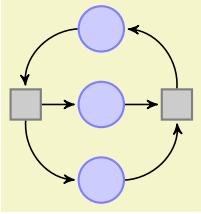
```
\begin{tikzpicture}
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical] {};
  \node[transition] (leave critical) [right of=critical] {};
  \node[transition] (enter critical) [left of=critical]  {};
  \draw [->] (enter critical) to                 (critical);
  \draw [->] (waiting)        to [bend right=45] (enter critical);
  \draw [->] (enter critical) to [bend right=45] (semaphore);
\end{tikzpicture}
```

是时候Hagen来了解另一种指定边缘的方法了:使用`edge`路径操作.这个操作与`to`非常相似,但是有一个很大的不同: 与结点一样,边缘是用`edge`操作来产生的,且它不是主路径的一部分,而是最后才被加上的.这看上去并不重要,但是却会产生漂亮的结果. 例如,每个边缘都可以有它自己的箭头,自己的颜色等等,而且所有的边缘可以被放在同一条路径中.这将允许Hagen把代码写成:



```
\begin{tikzpicture}
  \node[place]      (waiting)                              {};
  \node[place]      (critical)       [below of=waiting]    {};
  \node[place]      (semaphore)      [below of=critical]   {};
  \node[transition] (leave critical) [right of=critical]   {};
  \node[transition] (enter critical) [left of=critical]    {}
    edge [->]                (critical)
    edge [<-,bend left=45]   (waiting)
    edge [->,bend right=45] (semaphore);
\end{tikzpicture}
```

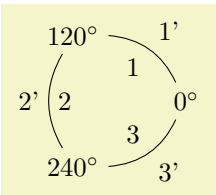
每一个edge会创建一条新的路径,这条路径是由to构成,它连接了结点enter critical与结点后edge命令. 最后一笔是要引入两个风格pre 和post,而且要用bend angle=45选项来设定弯曲角度.



```
% Styles place and transition as before
\tikzstyle{pre}=[<-,shorten <=1pt,>=stealth',semithick]
\tikzstyle{post}=[->,shorten >=1pt,>=stealth',semithick]
\begin{tikzpicture}[bend angle=45]
  \node[place]      (waiting)                              {};
  \node[place]      (critical)       [below of=waiting]    {};
  \node[place]      (semaphore)      [below of=critical]   {};

  \node[transition] (leave critical) [right of=critical]   {}
    edge [pre]               (critical)
    edge [post,bend right]   (waiting)
    edge [pre, bend left]    (semaphore);
  \node[transition] (enter critical) [left of=critical]    {}
    edge [post]              (critical)
    edge [pre, bend left]    (waiting)
    edge [post,bend right]   (semaphore);
\end{tikzpicture}
```

3.11 给线增加标签

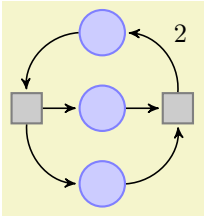
Hagen下面要做的一件事就是要在弧线处加上一个“2”.为此,他可以使用Ti*k*Z 的结点自动放置机制:通过增加选项auto, Ti*k*Z 将会把结点自动安放,它们会被放在直线或是曲线的边上而不会放在其上.增加swap选项将会把结点放在关于线镜面对称的地方. 这是一个一般的例子:




```
\begin{tikzpicture}[auto,bend right]
  \node (a) at (0:1) {$0^\circ$};
  \node (b) at (120:1) {$120^\circ$};
  \node (c) at (240:1) {$240^\circ$};

  \draw (a) to node {1} node [swap] {1'} (b)
        (b) to node {2} node [swap] {2'} (c)
        (c) to node {3} node [swap] {3'} (a);
\end{tikzpicture}
```

发生了什么?结点被放在to操作的内部!当你这样做了,那么结点将会被放在to操作创建曲线或直线的中间. auto选项然后就会发生作用,它把结点放在直线或是曲线的边上而不会放在其上.在这个例子中,我们甚至是每个to操作给出了两个结点.

对Hagen来说to操作倒不那么必须,因为两个“2”标签也能简单的“手工”放置.但是,在一个带有大量边缘的复杂的图形中自动安放机制可是一个大大的福音.



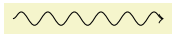

```
% Styles as before
\begin{tikzpicture}[bend angle=45]
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical] {};

  \node[transition] (leave critical) [right of=critical] {}
    edge [pre]                                  (critical)
    edge [post,bend right] node[auto,swap] {2} (waiting)
    edge [pre, bend left]                       (semaphore);
  \node[transition] (enter critical) [left of=critical]  {}
    edge [post]                                 (critical)
    edge [pre, bend left]                       (waiting)
    edge [post,bend right]                      (semaphore);
\end{tikzpicture}
```

3.12 添加蛇型线和多行文本

通过结点机制,Hagen现在可以轻而易举的创建两个Petri nets图.但是他对如何在网图中创建蛇型线并不太清楚.

为此,他可以使用一个*snake*.蛇型线这所以这样叫它,是因为大部分基本的蛇型线看上去确实比较像蛇.但是有些重复的样式也可以看成蛇型线,如振荡波状或是锯齿状或是其它一些更加复杂的形状.

要画出蛇型线,Hagen只需要给路径加上`snake=snake`选项.这将会导致路径上的所有的直线变成蛇型线.当然也可以只在路径的某一部分使用蛇型线,但这里Hagen用不上.

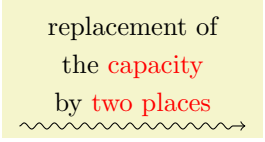


```
\begin{tikzpicture}
  \draw [->,snake=snake] (0,0) -- (2,0);
\end{tikzpicture}
```

好的,但那看上去不太对头.问题出在蛇型线的末端直接跟上的就是箭头.幸运的是,有一个选项可以避免它发生. 同时,蛇型线可以更小,这可以使用更多的选项来得到.


```
\begin{tikzpicture}
  \draw [->,snake=snake,
         segment amplitude=.4mm,
         segment length=2mm,
         line after snake=1mm] (0,0) -- (3,0);
\end{tikzpicture}
```

现在Hagen需要在蛇型线上方增加文本了.这看上去挺有挑战性的,因为这是一个多行文本. 要排版这些文本,Hagen需要指定文本的宽度同时指定这些文本要居中排列.

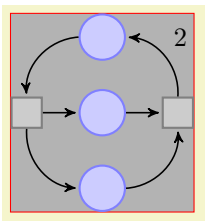



```
\begin{tikzpicture}
  \draw [->,snake=snake,
         segment amplitude=.4mm,
         segment length=2mm,
         line after snake=1mm] (0,0) -- (3,0)
    node [above,text width=3cm,text centered,midway]
    {
      replacement of the \textcolor{red}{capacity} by
      \textcolor{red}{two places}
    };
\end{tikzpicture}
```

3.13 使用层:背景矩形

Hagen还要添加背景矩形.这有点棘手:Hagen希望在画完Petri nets图之后再画出这些矩形.原因是:只有Petri nets图作出之后才能方便的指定矩形的顶点坐标.如果Hagen先画出这个矩形,那么他必须要知道Petri nets图的精确尺寸–而这是不可能的.

解决的方法是使用层.一旦背景库被载入,Hagen就可以把他的图的某部分放在一个`{pgfonlayer}`环境中. 然后这部分图就成为该环境参数指定的层的一部分.当`{tikzpicture}` 环境结束,这些层将会从背景层开始,由下而上一层一层的铺上去.




```
% Styles as before
\begin{tikzpicture}[bend angle=45]
  \node[place]      (waiting)                          {};
  \node[place]      (critical)       [below of=waiting]  {};
  \node[place]      (semaphore)      [below of=critical] {};

  \node[transition] (leave critical) [right of=critical] {}
    edge [pre]                                  (critical)
    edge [post,bend right] node[auto,swap] {2} (waiting)
    edge [pre, bend left]                       (semaphore);
  \node[transition] (enter critical) [left of=critical]  {}
    edge [post]                                 (critical)
    edge [pre, bend left]                       (waiting)
    edge [post,bend right]                      (semaphore);

  \begin{pgfonlayer}{background}
    \filldraw [fill=black!30,draw=red]
                (semaphore.south -| enter critical.west)
      rectangle (waiting.north   -| leave critical.east);
  \end{pgfonlayer}
\end{tikzpicture}
```

3.14 完整的代码

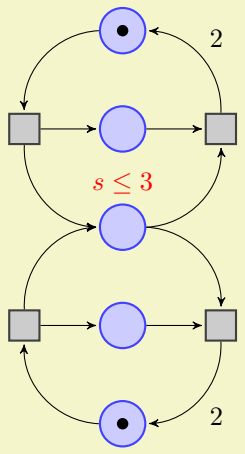
Hagen现在终于完成了所有的事情.只不过他并不知道早已有一个库用来画出Petri nets图!这个库基本上是提供了与Hagen 所完成事情的相同的定义.例如,它定义了一个`place`风格,与Hagen的方法很类似.调整一个代码,使用库来简化一点Hagen的代码后,如下.

首先,Hagen需要更少的风格定义,但是他仍需要指定位置和过渡[8] 的颜色

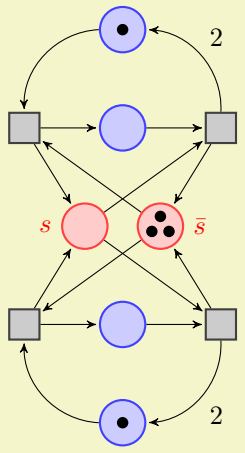
```
\tikzstyle{every place}=      [minimum size=6mm,thick,draw=blue!75,fill=blue!20]
\tikzstyle{every transition}=[thick,draw=black!75,fill=black!20]

\tikzstyle{red place}=        [place,draw=red!75,fill=red!20]

\tikzstyle{every label}=      [red]
\begin{tikzpicture}[node distance=1.3cm,>=stealth',bend angle=45,auto]
```

现在这个网图的代码为:

[8] Petri nets术语,原文为places and transitions




```
\node [place,tokens=1] (w1)                                  {};
\node [place]          (c1) [below of=w1]                    {};
\node [place]          (s)  [below of=c1,label=above:$s\le 3$] {};
\node [place]          (c2) [below of=s]                     {};
\node [place,tokens=1] (w2) [below of=c2]                    {};

\node [transition] (e1) [left of=c1] {}
  edge [pre,bend left]                 (w1)
  edge [post,bend right]               (s)
  edge [post]                          (c1);
\node [transition] (e2) [left of=c2] {}
  edge [pre,bend right]                (w2)
  edge [post,bend left]                (s)
  edge [post]                          (c2);
\node [transition] (l1) [right of=c1] {}
  edge [pre]                           (c1)
  edge [pre,bend left]                 (s)
  edge [post,bend right] node[swap] {2} (w1);
\node [transition] (l2) [right of=c2] {}
  edge [pre]                           (c2)
  edge [pre,bend right]                (s)
  edge [post,bend left]  node {2}      (w2);
```




```
\begin{scope}[xshift=6cm]
  \node [place,tokens=1]     (w1')                              {};
  \node [place]              (c1') [below of=w1']               {};
  \node [red place]          (s1') [below of=c1',xshift=-5mm]
          [label=left:$s$]                                      {};
  \node [red place,tokens=3] (s2') [below of=c1',xshift=5mm]
          [label=right:$\bar s$]                                {};
  \node [place]              (c2') [below of=s1',xshift=5mm]    {};
  \node [place,tokens=1]     (w2') [below of=c2']               {};

  \node [transition] (e1') [left of=c1'] {}
    edge [pre,bend left]                  (w1')
    edge [post]                           (s1')
    edge [pre]                            (s2')
    edge [post]                           (c1');
  \node [transition] (e2') [left of=c2'] {}
    edge [pre,bend right]                 (w2')
    edge [post]                           (s1')
    edge [pre]                            (s2')
    edge [post]                           (c2');
  \node [transition] (l1') [right of=c1'] {}
    edge [pre]                            (c1')
    edge [pre]                            (s1')
    edge [post]                           (s2')
    edge [post,bend right] node[swap] {2} (w1');
  \node [transition] (l2') [right of=c2'] {}
    edge [pre]                            (c2')
    edge [pre]                            (s1')
    edge [post]                           (s2')
    edge [post,bend left]  node {2}       (w2');
\end{scope}
```

背景与蛇形线的代码如下:

```
  \draw [-to,thick,snake=snake,segment amplitude=.4mm,segment length=2mm,line after snake=1mm]
    ([xshift=5mm]s -| l1) -- ([xshift=-5mm]s1' -| e1')
    node [above=1mm,midway,text width=3cm,text centered]
      {replacement of the \textcolor{red}{capacity} by \textcolor{red}{two places}};

  \begin{pgfonlayer}{background}
    \filldraw [line width=4mm,join=round,black!10]
      (w1.north  -| l1.east)  rectangle (w2.south  -| e1.west)
      (w1'.north -| l1'.east) rectangle (w2'.south -| e1'.west);
  \end{pgfonlayer}
\end{tikzpicture}
```

4 制图指南

这一章跟PGF 或TikZ 无关,如何创建学术性幻灯片,论文和书籍中图形,本章给出了一些方针与原则.

这篇指南的灵感来自于不同的地方.其中大部分我认为是“常识”,另一些则是我的个人经验总结(而不是个人偏好),还有一些来自于其它书本(不好意思,参考文献还没有完成)中有关图形设计和排版的内容. 其中给我灵感最多的是Edward Tufte的一本伟大的书.我并非完全赞同书中所说的一切,无论怎样,他书中的很多观点都是很有说服力的,我将在下面的指南中再次提及这些观点.

当有人把一本指南放在你的面前时,你要问自己的第一件事恐怕是:我真的要按它说的做吗?这是一个重要的问题,因为也有很好的理由不去遵守这些指南.

- 写一篇使用指南的人,他所面对的目标不止你一个.例如,一个一个指南可能这样写“用红色来表示强调”.这个指南则想当然的认为:在你用黑白色打印时那么用红色具有*相反的*效果,才会有强调的作用.

  指南也常常面对一定的情形.如果你不属于这种情形,那么按指南上说的做,则弊多利少.

- 排版中最基本的规则是:“任何规则都可以被打破,当你清醒的知道你正在打破规则”.这条规则同样也适用于图形. 换句话说,最基本的规则是:“排版中你唯一会犯的错误是:对你所做的事的无知”.

  当你已经足够了解这条规则,且你也认为只有打破规则才会得到你要的效果,那么,不要犹豫,去打破这个规则吧.

所以,在你要采纳或不采纳这个指南之前,先问一下自己:

1. 这个指南真的适合我的情形吗?
2. 如果你按照指南所说的相反的方法去做,那么你获得的好处是否多于那些弊端呢?这些弊端本来是可以用指南中的方法来避免发生的.

4.1 计划作图所需要的时间

当你要创建一个有大量图片的论文,那么创建它们所要花费的时间则是一个不可小视的因素.在创建图形上总共要花费多少时间呢?

一个一般的规则,认为在图片上花费的时间应该与创建相当长度文本的时间大体一致.例如,我写论文时,第一个草稿我每页需要大约一个小时.后来,我将会花费每页二到四小时来校正.这样,我需要用半个小时来创建半页图形的**第一个草图**.再下来,我会花费另一到两个小时,直到完成最后的图形.

在很多的出版物,甚至一些好的杂志中,作者和编者很明显投入了大量时间在文字写作上,但是好像只花了五分钟来创建所有的图形. 图形往往是因为“事后才想到”才被增加入文章,而有的图形仅仅只是作者的那些统计软件所给结果的一个屏幕截图.而且一些程序(如: GNUPLOT)给出的图形,其默认的输出品质很糟糕.

创建一个便于帮助读者理解文章的,适合主旨,信息丰富的图形是一件艰难而冗长的过程.

- 把图形当成你的论文的一等公民.它们值得花费与文字同样的时间和精力.
- 不可争辩的,创建图形理应花费比写文本更多的时间,因为人们总是更注意图片,而且总是先看到图片.
- 为创建和校正一个图形付出足够的时间,时长应与写下相同文本所花的时间一样.
- 含有高密集信息的难图,可能需要更多的时间.
- 简单的图需要很少的时间,但是你可能不会想在你的论文中放置一个“非常简单”的图; 就像你也不会在你的论文中放置一个相同尺寸的“很简单的文字”一样.

4.2 创建图形的流程

当你写一个(科技)文章,你最好遵循以下的模式:你有些成果或想法需要让人知道.论文的写法一般是先组织一个大体的轮廓.然后, 写出不同的章节以便创建第一个草稿.这个草稿然后会一直被修订,经常是多次的校正,直到最后论文成形.一篇好的杂志论文,最后应该是初稿中没有一句话是没有被修改过的.

创建一个图形遵循相同的模式:

- 确定图形想要传递什么.对此要有清醒的认识,就是说,决定“这个图形要告诉读者的是什么?”
- 建立一个“轮廓”,就是说,一个粗略的图形的总“外形”,它要包含图形最关键的元素.通常, 使用铅笔或是钢笔是会帮很大忙的.
- 补充上更多的细节,来创建第一个草图.
- 修订图形,直至论文完工.

4.3 把图形与文章主体联系起来

图形可以放大文本中的不同地方.可以把它们放在行间,就是说放在“文本的是间”,也可以把它们单独放置. 由于打印者(人们)希望页面要放的“满满”的(既美观又经济),单独放置的图形常常会与它们所关联的主题内容距离甚远. LaTeX 和TeX 也常常出于技术层面的原因,鼓励这种“浮动形式”.

如果那是一个行内图形,那么它或多或少的被视为自动连向当前文本,因为感觉上我们默认总是以其周围的文字来描绘这个图形的. 同时,当前文本也通常是用来解释图形是关于什么展示什么的.

不同的是,一个单独放置的图形,用以解释说明其的文本则有可能是前面你已经读到过的,亦或是你尚未读到的.出于这个原因,你在创建单独放置图形时,应该遵循以下方针:

- 单独放置图形应该有一个标题,甚过让他们可以“不言自明”

  举个例子,假设一个图形展示了一个快速排序算法的不同阶段.那么图形的标题最起码应该标上“这个图形展示了快速排序算法的不同阶段,内容见xyz页”,而不能只标上“快速排序算法”

- 一个好的标题应该要尽量增加信息量.举个例子,你可以说:“这个图形展示了快速排序算法的不同阶段,内容见xyz页.在第一条线上,选中主元5,这将导致...”当然,这个信息也可以在主文本中给出,但是把它放在图形的标题中会使图形的来龙去脉更加清晰.不要害怕一个五行标题.(你的编辑可能比较恨于这种情形.那么你可以考虑恨回去)

- 在你的文本中引用这个图形,如“一个关于快速排序是如何’执行’的例子,见图 2.1 第xyz 页”

- 大量的关于风格和排版的书建议你不要使用缩写,如不要写成“Fig. 2.1”而是写为“Figure 2.1.”[9]

  不使用缩写的主要论点在于”句号太重要了,不能浪费在缩写上”.他们这样认为：一个句号会使读者认为句子在“Fig”后面就结束了. 让人们在“有意识的回溯”后才会认识到这个句子并没有真的结束.

  支持使用缩写的论点的出发点是,它们节约空间.

  个人看来,我对两个论点不置可否,因为一方面,没有任何证据显示缩写减慢了读者的阅读速度;另一方面,把所有的“Figure”缩写成“Fig.”在多数文件中并不能节省哪怕是一张纸.

  本人就是避免使用缩写.

[9] 译者注:中文没有这个问题

4.4 图形与文字的一致性

人们在作图的时候最易犯的一般性“错误”(记住:设计中的错误往往就是无知的表现)可能就是图形的内容与文章的主题不配对.

作者使用多种不同的程序来为自己的论文作图,这种现象很普通.一个作者可能使用GNUPLOT来制作函数图,或是使用XFIG 来绘制图表,也可能会引入由另一个合作作者用不知名程序绘制的某个.eps图形.通常情形下,所有的这些图可能会使用不同的线粗, 不同的字体,不同的尺寸.另外,引入图形时作者们为了使之放缩到一个“漂亮的尺寸”,经常会使用像[height=5cm]这种类型的选项.

如果在写文章中也使用同样的方法,那么每一节的标题将会写成不同的字体不同的尺寸,在某些章中所有的定理都被加上下划线,在某些章它们会被打印成大写,而在另某些章会被打印为红色.另外,每一页的页边距也不一样.

如果一个文章写成这个样子,读者和编辑们将会难以忍受,但是对于图形,他们却不得不忍受一下.

要使得图形与文本的一致,坚持以下原则:

- 不要缩放图形.

  也就是说,当你用一个外部程序来作图的时候,那尽量把它们生成为“正好的尺寸”.

- 在图形和主体文章中使用相同的字体.

- 在图形和主体文章中使用相同的线粗.

  普通文本的线粗是字母T中竖线的宽度.对于 TEX 来说,通常是0.4 pt.但是,有些杂志不会接受线粗小于0.5 pt的图像.

- 当使用颜色时,对在文字和图形中使用一致的颜色代码.例如,如果假设红色是用来提醒读者文本中重要内容的话,那么在图形中重要内容也要用红色可标注.如果蓝色被用来表示结构部件如大标题和章名,那么你的图形中的结构部件也要用蓝色.

  但是,图形也可以使用一个合乎逻辑的内嵌颜色代码.例如,不管你合理的使用什么颜色,读者总是这样认为“绿色代表‘积极,前进,好’, 红色总是代表‘警报,警告,行动”’

当使用不同的作图程序时尽量的保持其一致性.出于这个原因,你应该考虑一下始终使用同一个作图程序.

4.5 图像的标签

几乎所有的图形都包含标签,标签其实就是对图形的某个部分进行说明的文字.当放置标签的时候,遵守以下的指南:

- 放置标签的时候遵循一致性原则.包含以下两个方面:首先,与主体文章保持一致,也就是说标签要使用与主体文章一样的字体. 第二,标签之间也要一致,也就是说,如果你的某一个标签采用了某种格式,其它所有的标签也要采用相同的格式.

- 除了要对文本和图形使用相同的字体之外,你也应当使用相同的标记法.例如,如果你在文章中用1/2,那么在图形的标签中也要使用1/2,而不是“0.5”.π就是π,而不是“3.141”.最后, $e^{-i\pi}$就是“$e^{-i\pi}$”, 不是“-1”, 不管“-1”的事

- 标签必须清晰.它们不光要具人合理的大小,还要不能被其它线或是文本所遮盖,这也同样适用于标签后面的线和文本.

- 标签应该“在其位”.不管是否有足够的空间,标签总是要放在它们所要标识处的边上.只在某些必要的时候,增加一条从标签到所标示物的(轻淡的)线.应该避免仅仅引用外部图示的解释说明,这样,读者必须让目光从图形和其解释之间来回跳跃.

- 考虑用轻淡的颜色来加上”不重要”的标签,例如灰色.这将会焦点集中在图形上.

4.6 函数图像和图表

最常见的一种图形格式,恐怕就是函数图像了,尤其在科技文章中.各种图像种类繁多,包括:简单直线图像,参数方程图像,三维图像,饼状图表等等.

不幸的是,图像的绘制众所周知的难于驾驭.某些程序如GNUPLOT或Excel的默认设定常常为此而备受责难.因为这类程序可以非常方便的创建不好的图像. 当你要创建一个函数图像时,你要问的第一个问题是:

- 有足够多的点来创建图像吗?

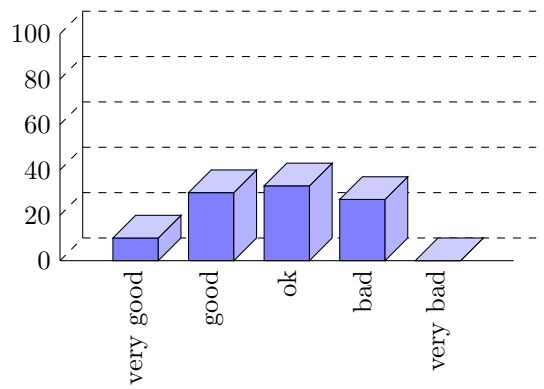
如果答案是“确实没有”,那就用使用表格.

一个典型情形是,当你要用一个条状图来展示些许数据, 那么没有必要使用图像这里有一个真实的例子:某个讨论会的结束后,一个演讲者请其他参与者给于反馈.共50个参与者中,有30个填写了反馈表格.根据反馈表,有3名参与者认为讨论会“很好”,9名认为“好”,10名“一般”,8名“糟糕”,没有人认为是“很糟糕”

把这些数据汇总在一起的一个简单的方法就是使用下面的表格:

Rating given	*Participants (out of 50) who gave this rating*	*Percentage*
“very good”	3	6%
“good”	9	18%
“ok”	10	20%
“bad”	8	16%
“very bad”	0	0%
none	20	40%

这位演讲者的做法是用一个3D 条图来使之更加具体化,如下:



在此,表格和“图像”具有差不多的尺寸.如果你首先的想法是“图像比表格漂亮”,那么请你基于表格或是基于图像来回答以下几个问题:

1. 总共有多少个参与者?
2. 有多少个参与者填写了反馈表?
3. 填写反馈表的参与者的比例?
4. 多少参与都认为“很好”?
5. 选中“很好”的参与者占总数的比例?

6. 选中“糟糕”或是“非常糟糕”的参与者超过总参与者的四分之一吗?

7. 填写了反馈表的参与者中选择“很好”的比例

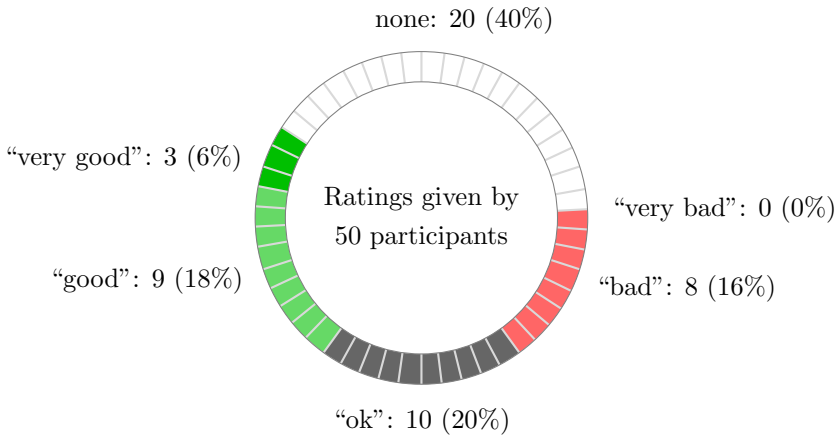
很遗憾,图像不能让我们回答*其中任何一个问题*.表格直接回答了所有问题,除了最后一个. 实际上,这个图像的信息密集度几乎为零.这个表格具有高度的信息密集度,尽管事实上它用了一大块空白空间来展示不多的数据.

下面列出了3D条状图易出毛病的几件事:

- 整个图像充满了烦人的背景线条.
- 左边的数据含义不清;大概是百分比,也有可能是参与者的具体数目.
- 下面的标签是旋转过的,不便阅读.

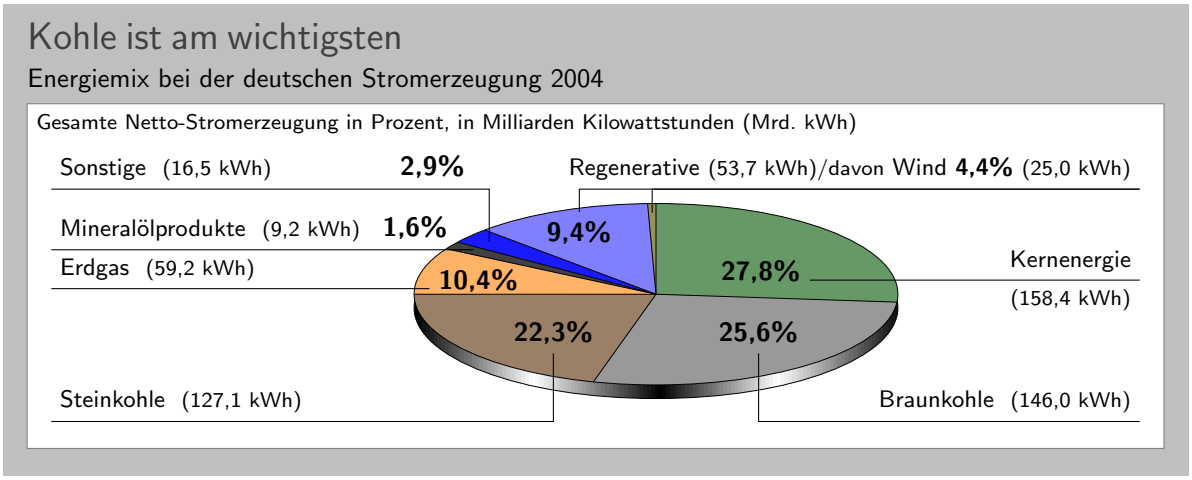
  (在我看到过的一个真实的幻灯片中,这个文字采用了每个字母10乘6的低分辨率,且字间距也不对,事实上造成了这个旋转过的文字根本无法阅读)
- 三维的效果以图像增加了复杂度却没有增加信息量.
- 三维的设定使得对柱条的高度很难估测.看一下“糟糕”柱条.这个柱条所代表的数值是多于20还是少于20?图上看来,柱条的前端在20这条线的下方,柱条的后端(它的读数)却在线的上方.
- 要分清这些柱条分别代表哪些数字几乎不可能.这样,这些柱条就毫无必要的掩藏了这样一个信息,即: 它们是要表述什么的.
- 这些柱条的高度这和是多少?是100% 还是60%?
- “很糟糕”所指柱条代表的是0还是1?
- 为什么这些柱条要是蓝色?

你也许会争辩称,在这个例子中精确的数字对于图像来说并不重要.重要的事情是这样的“讯息”,即认为“非常好”和“好”所占的比例要比认为“糟糕”和“非常糟糕”所占比例大. 但是,要传递这个讯息可以用一句话直接说出来,或者是用一个使信息传递更清晰的一个图:



上面的图与那个表格具有差不多的信息密集度(大约相同的尺寸,展示了相同的数据).另外,人们可以直接“看出” “非常好”和“好”所占的比例要比“糟糕”所占比例大.人们也可以直接“看出”没有填写反馈表的人的数目, 他们并没有被忽视,这在反馈表的填写中非常普遍.

图表时常并不是一个好的办法.让我们看一个例子,这是我对2005年6月4号的《Die Zeit》中的一个饼状图的重画.



这个图像用了Ti*k*Z来重画,但是看上去和原版差不多.
第一眼看上去,这个图像看上去“漂亮而富有信息”,但是也有很多的地方不尽如人意:

- 这个图表是三维的.但是其底纹并没有增加“信息导向”,它们最多是分散了注意力.
- 在一个3维饼状图表中,相关的尺寸被大大的扭曲了. 例如,此图中表明“Braunkohle”的灰色部分面积比表明“Kernenergie”的绿色部分面积要大*尽管事实上Braunkohle 的百分比要比Kernenergie 来的小*
- 3D失真使小面积区域的情形更加糟糕.“Regenerative”区域要比“Erdgas”区域大一些.“Wind”区域要比“Mineralölprodukte”区域小一点,*尽管Wind 的百分比大概有Mineralölprodukte百分比的三倍大*

  在最后一个例子中,尺寸的不同仅仅有一部分原因归咎于失真.原始图形的设计者(们)也使“Wind”薄片变的太小,考虑失真的原因后依然太小.(对比一下“Wind” 和“Regenerative”)
- 从它的标题可知,这个图表是要告诉我们200年德国最重要的能源是煤.我们要克服那些多余的迷惑人的3维设置所带来的干扰,需要一段时间才能搞明白它要表明的意思.

  作为能源的煤被分隔成两个薄片:一个是“Steinkohle”,另一个是“Braunkohle”(两种不同各类的煤).当你把它俩加起来,你会发现这个饼状图的将近一半被煤所占据.

  表示两个煤的两块区域没有任何视觉上的联系.相反的,使用了两种不同的颜色,标签也被放在了图的两边.通过对比, “Regenerative”和“Wind”却非常紧密的连在一起.
- 此图的颜色使用也无任何逻辑可言.为什么核能要用绿色?再生能源用淡蓝色,“其它能源”用蓝色.而“Braunkohle”(字面意思为棕色煤) 用了石头灰色同时“Steinkohle”(字面意思为石头煤)用了棕色,更像是作者跟大家开了个玩笑.
- 颜色最亮的区域是“Erdgas”.这个区域如此突出主要是因为使用了最亮的颜色.但是,在这个图表中,“Erdgas”实际上一点也不重要.

Edward Tufte把上面这类图称为“图表垃圾”.
这里有一些建议可以帮助你避免产生图表垃圾.

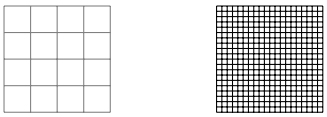
- 不要使用3维饼状图.它们是魔鬼.
- 考虑用一个表格来代替饼状图.
- 不要随机的乱用颜色;要用颜色来引导读者的注意力,用它们对事情分类.
- 不要用背景图案,如平行线相交阴影或对角线,改用颜色.信息图表中的背景图案是魔鬼.

4.7 集中与分散注意力

拿出一本你最爱读的科幻小说,并翻开其中的某一页,你会发现此页是非常.没有什么会分散读者阅读时的注意力,没有大标题,没有黑体字,没有大片空白.事实上,就算作者要强调什么,他会使用斜体字. 这种字体会与其他文本完美的混合在一起—-一个页面是否包含斜体字其距离的变化你根本分辨不出来,但是如果是一个黑体字,你一眼就会看出来.小说之所以用这种方式排版,是因为要遵循以下的模式:避免分散注意力好的排版(如一些优秀的杂志)就是你不会注意到的排版.排版的任务就是让人来读这些文字,也就是说, 让读者尽可能不费力的“吸收”文章的信息量.对小说来说,读者通过一行一行的阅读来获知其内容,就像他们听某个人说故事一样.这种情形下,页面上干扰眼睛快速而平稳的进行行间扫描的任何事都会增加阅读的难度. 现在,拿出你最喜欢的周刊或报纸,看一下其中的某一页.你会发现,页面上有很多事情“发生着”. 字体以不同的大小出现,并安排在不同的地方,文本放在狭窄的列中且常常是文图交错. 这些刊物以这种方式排版是出于要遵循另一种模式:引导注意力. 读者不会像读小说一样去读一本杂志.我们不会去逐行的读一本杂志,相反地,我们以大标题和短的摘要来判断我们是否要去阅读某篇文章.这种排版的任务就是先把我们的注意力引导到这些摘要和大标题上. 一旦我们下定决心去读一篇文章,我们将难以再忍受这种注意力分散,这恰恰是文章的主体部分以小说的方式去排版的原因. 这两个原则“避免分散注意力”和“引导注意力”也适用于图形.当你要设计一个图形时,你应该剔除一切“分散眼球”的东西.同时,你应该通过使用字体/颜色/线条宽度,来高亮显示不同的部分,去试着积极的帮助读者“看透图形”.

下面是一个不完全列表,列出了会分散读者注意力的一些事.

- 高对比度的总是先被双眼所记录.例如,考虑下面两个网格:

  尽管按照正常的阅读顺序会先看到左边的网格,但更可能是先看到右边的:白黑对比要比灰白对比强烈. 另外,右边的网格中还有其它的“地方”增加了总体的对比度.

  像网格线之类(更一般的是辅助线)通常不应该去吸引读者的注意力,因此要把它们排版成与背景成低对比度. 同样,一个松散的网格线要比一个密度的网格线更不会分散注意力.

- 虚线中含有大量的黑白对比的点.虚线或是点线可能会非常的分散注意力,因此一般情况下应该避免使用.

  不要以不同的虚线模式用来区分函数图像中的曲线.你松散了图形上的这些数据点,而且人眼并不擅长于“以虚线的样式来区分事物”. 人眼更擅长于以颜色的不同来区分事物.

- 充满斜线、水平与竖直线或点线的填充背景常常是分散注意的,而且通常不会达到效果.

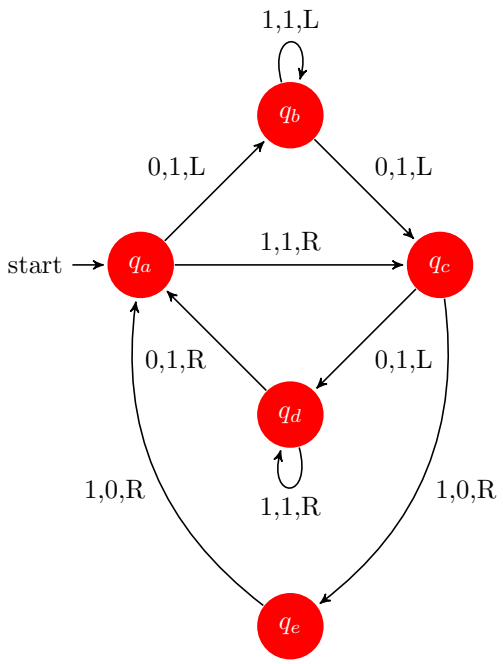
- 背景图片或阴影分散注意,而且几乎不会增加图形的任何重要信息.

- 可爱的小剪切画很容易把你的注意力从数据上转移走.

第二部分 II

安装和配置

这一部分解释了如何安装系统.一般情况下,有人已经为你安装好了,这时你可以略过这部分;但是如果没有安装好,或者如果你就是那个负责安装的可怜家伙的话,就请阅读此部分.




```
\begin{tikzpicture}[->,>=stealth',shorten >=1pt,auto,node distance=2.8cm,semithick]
  \tikzstyle{every state}=[fill=red,draw=none,text=white]

  \node[initial,state] (A)                    {$q_a$};
  \node[state]         (B) [above right of=A] {$q_b$};
  \node[state]         (D) [below right of=A] {$q_d$};
  \node[state]         (C) [below right of=B] {$q_c$};
  \node[state]         (E) [below of=D]       {$q_e$};

  \path (A) edge              node {0,1,L} (B)
            edge              node {1,1,R} (C)
        (B) edge [loop above] node {1,1,L} (B)
            edge              node {0,1,L} (C)
        (C) edge              node {0,1,L} (D)
            edge [bend left]  node {1,0,R} (E)
        (D) edge [loop below] node {1,1,R} (D)
            edge              node {0,1,R} (A)
        (E) edge [bend left]  node {1,0,R} (A);
\end{tikzpicture}
```

5 安装

根据你的系统和你的要求,有不同的安装方式,你也许还需要安装其它的宏包,下面会提到. 在安装之前,你也许想要看一下宏包分发的许可证,见章节 6.

一般情况下,宏包已经安装在你的系统中了.自然的,你就完全不必去担心安装的过程了,你可以略过本章以下的部分.

5.1 宏包和驱动的版本

本文档是PGF 宏包1.10 版本的一部分.为了能够运行PGF,你的系统中需要有较新的TEX. 当使用LATEX 时,你要安装有以下的宏包(更高版本也可以):

- `xcolor`版本2.00.
- `xkeyval`版本1.8, 如果你要使用Ti*k*Z 的话.

如果使用plain TEX,`xcolor`不是必需的,但你显然就不能获得所有功能了.
目前,PGF 支持以下的后台驱动:

- `pdftex` 版本0.14 或更高.更早期的版本不能工作.
- `dvips`版本5.94a 或更高.更早期的版本也可能会工作.对于图像内部连接(inter-picture connections) 你需要`pdftex`版本1.40或以上,以DVI模式来处理图像.
- `dvipdfm` 版本0.13.2c或是更高.更早期的版本也可能会工作.对于图像内部连接(inter-picture connections) 你需要`pdftex`版本1.40或以上,以DVI模式来处理图像.
- `tex4ht` 版本2003-05-05 或更高.早期的版本也可能会工作
- `vtex` 版本8.46a 或更高.早期的版本也可能会工作.
- `textures` 版本2.1 或更高.早期的版本也可能会工作.

当前,PGF 支持以下格式:

- `latex`,全部功能
- `plain`,除了只能用于pdfTEX 的图形引用功能之外的全部功能
- `context`,除了只能用于pdfTEX 的图形引用功能之外的全部功能

更多细节,见章节 7.

5.2 安装集成包

我并没有创建或是管理任何PGF 的集成包,但是幸运的是,一些好心人帮我做了.因为我不管理这些集成包, 所以我无法以出详细的用法说明,但是我*可以*告诉你在哪可以找到这些说明.如果你遇到安装的问题, 你也许先要看一下Debian和MikTEX 的网页

5.2.1 Debian

命令"`aptitude install pgf`"为你安排好一切.坐下,休息一会.详细的来说,以下的包被安装了:

`http://packages.debian.org/pgf`
`http://packages.debian.org/latex-xcolor`

5.2.2 MiKTeX

对于MiKTEX 来说,使用更新向导安装(最新版本的)pgf, xcolor, 和xkeyval宏包.

5.3 安装在texmf目录树中

当你要求TEX 使用某一个类或是宏包时,它通常用在一个被称为texmf树的目录中去找必要的文件. 默认情况下,TEX 会在三个不同的texmf目录树中去寻找文件.

- 根texmf目录树,通常位于/usr/share/texmf/或c:\texmf\或类似的某处.
- 本地texmf目录树,通常位于/usr/local/share/texmf/ 或c:\localtexmf\或类似的某处.
- 你的个人texmf目录树,通常位于~/texmf/ 或~/Library/texmf/

你可以把包安装在本地目录树或是个人目录树,取决于你对本地目录树有无写权限. 安装在根目录树中可能会引起问题,因为一次TEX 系统的完全升级安装,可能就会替换回整个目录树的所有文件.

5.3.1 把包内文件安装在一起

一旦你找到了正确的目录树,你必须要决定是否把PGF 的所有文件装在一个地方,或是使用"TDS方式"安装文件,这里TDS指的是"TEX 目录结构"

如果你希望把所有的文件放在texmf目录树的一个地方,你要先建一个子子文件夹命名为texmf/tex/generic/pgf 或 texmf/tex/generic/pgf-1.10(如果你愿意的话).然后把pgf包中的所有文件都放在这个目录里.最后,刷新TEX 文件名数据库,可以使用命令texhash 或mktexlsr(它们其实是一样的).在MikTEX 中,有一个菜单选项也可以完成刷新.

5.3.2 TDS-方式安装

以上的安装过程是很"自然的"而且我也推荐这种方式,因为它使得升级或是管理PGF 包都很方便, 可是这种安装并不是"TDS方式"的.如果你想要"TDS方式"的,请用以下方法:(如果你不晓得"TDS方式" 是什么意思的话,你也许就不需要使用"TDS方式"了)

pgf包的.tar文件的根目录下包括以下文件和目录:README, doc, generic, plain,和latex.你要把这里四个目录每一个都"合并"到相应的目录中去,它们是:texmf/doc, texmf/tex/generic, texmf/tex/plain, 和texmf/tex/latex.例如,在.tar文件的doc目录中包含一个叫pgf的目录,你必须把这个目录移动到texmf/doc/pgf 中去.根目录下的README文件可以被忽略掉,因为它只是doc/pgf/README文件的一个复本.

你也想到把所有的文件放在一起并使用符号链接把文件指向"TDS方式"的对应目录

关于标准安装过程的更多详细说明你可以参考http://www.ctan.org/installationadvice/. 但请注意, PGF 宏包不包含.ins文件(只是省略了这一部分).

5.4 升级安装

要从前一个版本升级安装到最新版本的话,你只需把texmf/tex/generic/pgf目录下的所有文件有新版本的文件去代替就可以了(如果你用的是TDS方式安装,那就把pgf相关的所有目录都对应的替代). 简单的方法是先把旧版本删除再用以上说的方法安装一遍.有时,不同版本之间,一个命令的语法可能会有变化. 如果以前行得通的方法不再奏效,你最好得看一下发行说明和更正列表.

6 许可证和版权

6.1 用哪个许可证?

PGF 宏包的不同部分在不同的许可证下发行:

1. 此宏包的代码是双重许可证.这将意味着当你使用PGF 宏时可自己决定应用哪个许可证.这里有两个选择:
   (a) 你可以使用GNU Public License, version 2.
   (b) 你可以使用LaTeX Project Public License, version 1.3c.
2. 此宏包的文档也是双重许可证.同样,你可以从以下两者中任取其一:
   (a) 你可以使用GNU Free Documentation License, version 1.2.
   (b) 你可以使用LaTeX Project Public License, version 1.3c.

"此宏包的文档"是指pgf宏包中doc子目录下的所有文件.详细的文件列表可以在文件doc/generic/pgf/licenses/manifest
中找到.其它目录下的所有文件都是"此宏包的代码"的一部分.详细的文件列表可以在文件doc/generic/pgf/licenses/manifes
找到.

本节接下来将给出这些许可证.以下的文本是版权所有的,更多细节请看doc/generic/pgf/licenses目录下的纯文本文件.

本手册中的使用的示例图片(Brave GNU World的logo),是从Brave GNU World主页上获取,其版权声明如下:"Copyright (C) 1999, 2000, 2001, 2002, 2003, 2004 Georg C. F. Greve. Permission is granted to make and distribute verbatim copies of this transcript as long as the copyright and this permission notice appear."

6.2 The GNU Public License, Version 2

6.2.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its

recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

6.2.2 Terms and Conditions For Copying, Distribution and Modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves,

then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsubsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

6.2.3 No Warranty

10. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

11. In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

6.3 The LaTeX Project Public License, Version 1.3c 2006-05-20

6.3.1 Preamble

The LaTeX Project Public License (LPPL) is the primary license under which the the LaTeX kernel and the base LaTeX packages are distributed.

You may use this license for any work of which you hold the copyright and which you wish to distribute. This license may be particularly suitable if your work is TeX-related (such as a LaTeX package), but it is written in such a way that you can use it even if your work is unrelated to TeX.

The section 'WHETER AND HOW TO DISTRIBUTE WORKS UNDER THIS LICENSE', below, gives instructions, examples, and recommendations for authors who are considering distributing their works under this license.

This license gives conditions under which a work may be distributed and modified, as well as conditions under which modified versions of that work may be distributed.

We, the LaTeX3 Project, believe that the conditions below give you the freedom to make and distribute modified versions of your work that conform with whatever technical specifications you wish while maintaining the availability, integrity, and reliability of that work. If you do not see how to achieve your goal while meeting these conditions, then read the document '`cfgguide.tex`' and '`modguide.tex`' in the base LaTeX distribution for suggestions.

6.3.2 Definitions

In this license document the following terms are used:

Work Any work being distributed under this License.

Derived Work Any work that under any applicable law is derived from the Work.

Modification Any procedure that produces a Derived Work under any applicable law – for example, the production of a file containing an original file associated with the Work or a significant portion of such a file, either verbatim or with modifications and/or translated into another language.

Modify To apply any procedure that produces a Derived Work under any applicable law.

Distribution Making copies of the Work available from one person to another, in whole or in part. Distribution includes (but is not limited to) making any electronic components of the Work accessible by file transfer protocols such as FTP or HTTP or by shared file systems such as Sun's Network File System (NFS).

Compiled Work A version of the Work that has been processed into a form where it is directly usable on a computer system. This processing may include using installation facilities provided by the Work, transformations of the Work, copying of components of the Work, or other activities. Note that modification of any installation facilities provided by the Work constitutes modification of the Work.

Current Maintainer A person or persons nominated as such within the Work. If there is no such explicit nomination then it is the 'Copyright Holder' under any applicable law.

Base Interpreter A program or process that is normally needed for running or interpreting a part or the whole of the Work.

A Base Interpreter may depend on external components but these are not considered part of the Base Interpreter provided that each external component clearly identifies itself whenever it is used interactively. Unless explicitly specified when applying the license to the Work, the only applicable Base Interpreter is a 'LaTeX-Format' or in the case of files belonging to the 'LaTeX-format' a program implementing the 'TeX language'.

6.3.3 Conditions on Distribution and Modification

1. Activities other than distribution and/or modification of the Work are not covered by this license; they are outside its scope. In particular, the act of running the Work is not restricted and no requirements are made concerning any offers of support for the Work.

2. You may distribute a complete, unmodified copy of the Work as you received it. Distribution of only part of the Work is considered modification of the Work, and no right to distribute such a Derived Work may be assumed under the terms of this clause.

3. You may distribute a Compiled Work that has been generated from a complete, unmodified copy of the Work as distributed under Clause 2 above, as long as that Compiled Work is distributed in such a way that the recipients may install the Compiled Work on their system exactly as it would have been installed if they generated a Compiled Work directly from the Work.

4. If you are the Current Maintainer of the Work, you may, without restriction, modify the Work, thus creating a Derived Work. You may also distribute the Derived Work without restriction, including Compiled Works generated from the Derived Work. Derived Works distributed in this manner by the Current Maintainer are considered to be updated versions of the Work.

5. If you are not the Current Maintainer of the Work, you may modify your copy of the Work, thus creating a Derived Work based on the Work, and compile this Derived Work, thus creating a Compiled Work based on the Derived Work.

6. If you are not the Current Maintainer of the Work, you may distribute a Derived Work provided the following conditions are met for every component of the Work unless that component clearly states in the copyright notice that it is exempt from that condition. Only the Current Maintainer is allowed to add such statements of exemption to a component of the Work.

(a) If a component of this Derived Work can be a direct replacement for a component of the Work when that component is used with the Base Interpreter, then, wherever this component of the Work identifies itself to the user when used interactively with that Base Interpreter, the replacement component of this Derived Work clearly and unambiguously identifies itself as a modified version of this component to the user when used interactively with that Base Interpreter.

(b) Every component of the Derived Work contains prominent notices detailing the nature of the changes to that component, or a prominent reference to another file that is distributed as part of the Derived Work and that contains a complete and accurate log of the changes.

(c) No information in the Derived Work implies that any persons, including (but not limited to) the authors of the original version of the Work, provide any support, including (but not limited to) the reporting and handling of errors, to recipients of the Derived Work unless those persons have stated explicitly that they do provide such support for the Derived Work.

(d) You distribute at least one of the following with the Derived Work:

i. A complete, unmodified copy of the Work; if your distribution of a modified component is made by offering access to copy the modified component from a designated place, then offering equivalent access to copy the Work from the same or some similar place meets this condition, even though third parties are not compelled to copy the Work along with the modified component;

ii. Information that is sufficient to obtain a complete, unmodified copy of the Work.

7. If you are not the Current Maintainer of the Work, you may distribute a Compiled Work generated from a Derived Work, as long as the Derived Work is distributed to all recipients of the Compiled Work, and as long as the conditions of Clause 6, above, are met with regard to the Derived Work.

8. The conditions above are not intended to prohibit, and hence do not apply to, the modification, by any method, of any component so that it becomes identical to an updated version of that component of the Work as it is distributed by the Current Maintainer under Clause 4, above.

9. Distribution of the Work or any Derived Work in an alternative format, where the Work or that Derived Work (in whole or in part) is then produced by applying some process to that format, does not relax or nullify any sections of this license as they pertain to the results of applying that process.

10. (a) A Derived Work may be distributed under a different license provided that license itself honors the conditions listed in Clause 6 above, in regard to the Work, though it does not have to honor the rest of the conditions in this license.

(b) If a Derived Work is distributed under a different license, that Derived Work must provide sufficient documentation as part of itself to allow each recipient of that Derived Work to honor the restrictions in Clause 6 above, concerning changes from the Work.

11. This license places no restrictions on works that are unrelated to the Work, nor does this license place any restrictions on aggregating such works with the Work by any means.

12. Nothing in this license is intended to, or may be used to, prevent complete compliance by all parties with all applicable laws.

6.3.4 No Warranty

There is no warranty for the Work. Except when otherwise stated in writing, the Copyright Holder provides the Work 'as is', without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the Work is with you. Should the Work prove defective, you assume the cost of all necessary servicing, repair, or correction.

In no event unless required by applicable law or agreed to in writing will The Copyright Holder, or any author named in the components of the Work, or any other party who may distribute and/or modify the Work as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of any use of the Work or out of inability to use the Work (including, but not limited to, loss of data, data being rendered inaccurate, or losses sustained by anyone as a result of any failure of the Work to operate with any other programs), even if the Copyright Holder or said author or said other party has been advised of the possibility of such damages.

6.3.5 Maintenance of The Work

The Work has the status 'author-maintained' if the Copyright Holder explicitly and prominently states near the primary copyright notice in the Work that the Work can only be maintained by the Copyright Holder or simply that it is 'author-maintained'.

The Work has the status 'maintained' if there is a Current Maintainer who has indicated in the Work that they are willing to receive error reports for the Work (for example, by supplying a valid e-mail address). It is not required for the Current Maintainer to acknowledge or act upon these error reports.

The Work changes from status 'maintained' to 'unmaintained' if there is no Current Maintainer, or the person stated to be Current Maintainer of the work cannot be reached through the indicated means of communication for a period of six months, and there are no other significant signs of active maintenance.

You can become the Current Maintainer of the Work by agreement with any existing Current Maintainer to take over this role.

If the Work is unmaintained, you can become the Current Maintainer of the Work through the following steps:

1. Make a reasonable attempt to trace the Current Maintainer (and the Copyright Holder, if the two differ) through the means of an Internet or similar search.

2. If this search is successful, then enquire whether the Work is still maintained.

   (a) If it is being maintained, then ask the Current Maintainer to update their communication data within one month.

   (b) If the search is unsuccessful or no action to resume active maintenance is taken by the Current Maintainer, then announce within the pertinent community your intention to take over maintenance. (If the Work is a LaTeX work, this could be done, for example, by posting to `comp.text.tex`.)

3. (a) If the Current Maintainer is reachable and agrees to pass maintenance of the Work to you, then this takes effect immediately upon announcement.

   (b) If the Current Maintainer is not reachable and the Copyright Holder agrees that maintenance of the Work be passed to you, then this takes effect immediately upon announcement.

4. If you make an 'intention announcement' as described in 2b above and after three months your intention is challenged neither by the Current Maintainer nor by the Copyright Holder nor by other people, then you may arrange for the Work to be changed so as to name you as the (new) Current Maintainer.

5. If the previously unreachable Current Maintainer becomes reachable once more within three months of a change completed under the terms of 3b or 4, then that Current Maintainer must become or remain the Current Maintainer upon request provided they then update their communication data within one month.

A change in the Current Maintainer does not, of itself, alter the fact that the Work is distributed under the LPPL license.

If you become the Current Maintainer of the Work, you should immediately provide, within the Work, a prominent and unambiguous statement of your status as Current Maintainer. You should also announce your new status to the same pertinent community as in 2b above.

6.3.6 Whether and How to Distribute Works under This License

This section contains important instructions, examples, and recommendations for authors who are considering distributing their works under this license. These authors are addressed as 'you' in this section.

6.3.7 Choosing This License or Another License

If for any part of your work you want or need to use *distribution* conditions that differ significantly from those in this license, then do not refer to this license anywhere in your work but, instead, distribute your work under a different license. You may use the text of this license as a model for your own license, but your license should not refer to the LPPL or otherwise give the impression that your work is distributed under the LPPL.

The document '`modguide.tex`' in the base LaTeX distribution explains the motivation behind the conditions of this license. It explains, for example, why distributing LaTeX under the GNU General Public License (GPL) was considered inappropriate. Even if your work is unrelated to LaTeX, the discussion in '`modguide.tex`' may still be relevant, and authors intending to distribute their works under any license are encouraged to read it.

6.3.8 A Recommendation on Modification Without Distribution

It is wise never to modify a component of the Work, even for your own personal use, without also meeting the above conditions for distributing the modified component. While you might intend that such modifications will never be distributed, often this will happen by accident – you may forget that you have modified that component; or it may not occur to you when allowing others to access the modified version that you are thus distributing it and violating the conditions of this license in ways that could have legal implications and, worse, cause problems for the community. It is therefore usually in your best interest to keep your copy of the Work identical with the public one. Many works provide ways to control the behavior of that work without altering any of its licensed components.

6.3.9 How to Use This License

To use this license, place in each of the components of your work both an explicit copyright notice including your name and the year the work was authored and/or last substantially modified. Include also a statement that the distribution and/or modification of that component is constrained by the conditions in this license.

Here is an example of such a notice and statement:

```
%% pig.dtx
```

```
%% Copyright 2005 M. Y. Name
%
% This work may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3
% of this license or (at your option) any later version.
% The latest version of this license is in
%   http://www.latex-project.org/lppl.txt
% and version 1.3 or later is part of all distributions of LaTeX
% version 2005/12/01 or later.
%
% This work has the LPPL maintenance status 'maintained'.
%
% The Current Maintainer of this work is M. Y. Name.
%
% This work consists of the files pig.dtx and pig.ins
% and the derived file pig.sty.
```

Given such a notice and statement in a file, the conditions given in this license document would apply, with the 'Work' referring to the three files '`pig.dtx`', '`pig.ins`', and '`pig.sty`' (the last being generated from '`pig.dtx`' using '`pig.ins`'), the 'Base Interpreter' referring to any 'LaTeX-Format', and both 'Copyright Holder' and 'Current Maintainer' referring to the person 'M. Y. Name'.

If you do not want the Maintenance section of LPPL to apply to your Work, change 'maintained' above into 'author-maintained'. However, we recommend that you use 'maintained' as the Maintenance section was added in order to ensure that your Work remains useful to the community even when you can no longer maintain and support it yourself.

6.3.10 Derived Works That Are Not Replacements

Several clauses of the LPPL specify means to provide reliability and stability for the user community. They therefore concern themselves with the case that a Derived Work is intended to be used as a (compatible or incompatible) replacement of the original Work. If this is not the case (e.g., if a few lines of code are reused for a completely different task), then clauses 6b and 6d shall not apply.

6.3.11 Important Recommendations

Defining What Constitutes the Work The LPPL requires that distributions of the Work contain all the files of the Work. It is therefore important that you provide a way for the licensee to determine which files constitute the Work. This could, for example, be achieved by explicitly listing all the files of the Work near the copyright notice of each file or by using a line such as:

```
% This work consists of all files listed in manifest.txt.
```

in that place. In the absence of an unequivocal list it might be impossible for the licensee to determine what is considered by you to comprise the Work and, in such a case, the licensee would be entitled to make reasonable conjectures as to which files comprise the Work.

6.4 GNU Free Documentation License, Version 1.2, November 2002

6.4.1 Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

6.4.2 Applicability and definitions

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **"Document"**, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as **"you"**. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A **"Modified Version"** of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A **"Secondary Section"** is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **"Invariant Sections"** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **"Cover Texts"** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **"Transparent"** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose

markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called **"Opaque"**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **"Title Page"** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section **"Entitled XYZ"** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **"Acknowledgements"**, **"Dedications"**, **"Endorsements"**, or **"History"**.) To **"Preserve the Title"** of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

6.4.3 Verbatim Copying

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

6.4.4 Copying in Quantity

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

6.4.5 Modifications

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

6.4.6 Combining Documents

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6.4.7 Collection of Documents

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in

the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

6.4.8 Aggregating with independent Works

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

6.4.9 Translation

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

6.4.10 Termination

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

6.4.11 Future Revisions of this License

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

6.4.12 Addendum: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright ©YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with ... Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

7 输入和输出格式

TEX 被设计为一个灵活的系统.这不尽是对TEX 的*输入*格式而言,同样适用于它的*输出*格式. 这一节用来解释有哪几种输入格式,及PGF 是如何支持它们的.这里也解释了可以输出哪几种格式.

7.1 支持的输入格式

TEX 并没有规定你的输入格式.但是它有一些习惯用法,比如,一对花括号所包含的是TEX 的一个域,但并非是必要条件. 同时,它还有习惯用法如环境要以`\begin`开始,但是TEX 并不真的关心这些命令的名字.

尽管TEX 可以重新订制,但是一般用户没有这个能力.出于这个原因,某一个*输入格式*指定了一系列的命令和习惯来规范TEX 的格式.现在有三种"主要"的格式:Donald Knuth的原始的`plain` TEX 格式,Leslie Lamport的流行的LATEX 格式, 和Hans Hangen的ConTEXt 格式.

7.1.1 使用LATEX 格式

LATEX 中使用PGF 和Ti*k*Z 非常的方便:你只需声明`\usepackage{pgf}`或`\usepackage{tikz}`.通常,你只需做这些就够了, 所有的配置会自动完成并且正确无误(但愿).

LATEX 格式的类型文件放在PGF 系统的子目录`latex/pgf/`中.这些文件主要的功能是引用目录`generic/pgf`中的文件. 例如,下面是文件`latex/pgf/frontends/tikz.sty`的内容:

```
% Copyright 2005 by Till Tantau <tantau@users.sourceforge.net>.
%
% This program can be redistributed and/or modified under the terms
% of the GNU Public License, version 2.

\RequirePackage{pgf,calc,pgffor,pgflibraryplothandlers,xkeyval}

\input{tikz.code}

\endinput
```

`generic/pgf`目录中的文件,真正的去完成任务.

7.1.2 使用Plain TEX 格式

当使用plain TEX 时,你要声明`\input{pgf.tex}` 或`\input{tikz.tex}`.然后,你要使用`\pgfpicture`和`\endpgfpicture`.

与LATEX 格式不同的是,PGF 并不擅长完成plain TEX 的适当配置.特别情况下,只有当你使用`pdftex`或`tex` 加`dvips` 时,它才能自动的决定正确的输出格式.所有其它的输出格式,你需要设置宏`\pgfsysdriver`中的正确值.参阅后面要的输出格式的描述.

PGF 原本为使用LATEX 而写的,我已多次提到.但是,它对plain TEX 的支持也非常不错.

如同LATEX 的类型文件一样,plain TEX 的类型文件`tikz.tex`也同样只用来引用正确的`tikz.code.tex`文件.

7.1.3 使用ConTEXt格式

当使用ConTEXt格式,你需声明`\usemodule[pgf]` 或`\usemodule[tikz]`.如同plain TEX 一样,你也要把相应的环境开始环境结束标志改为:`\startpgfpicture` 和`\stoppgfpicture`;类似的,使用`\starttikzpicture` 和`\stoptikzpicture`,其它环境大约如此.

对ConTEXt的支持与plain TEX 的支持差不多,所有也有相同的限制:你可能不得不直接设定输出格式,而且图形的包含可能有问题.

另外,`pgf` 和`tikz` 也存在一起模块如`pgfcore`, `pgfbaseimage`, `pgflibrarysnakes`等等.要使用它们,你可能要先引入模块`pgfmod`(`pgf` 和`tikz`都包含有`pgfmod`模块,因此一般情况下,你可以略过这一步).这个特殊的模块是必

要的,因为ConTEXt 恶魔般地限制了模块的名字不能超过6个字母,PGF 的长名字要用模块pgfmod来映射成含义模糊的6字母名字.

7.2 支持的输出格式

所谓输出格式就是TEX 按排版要求输出的格式.产生输出(概念上)是一个两步走的过程:

1. TEX 排版出你的文字和图形.这个排版过程的结果主要是一个长的列表,这个列表由"字母–坐标"对加上(可能的)一些"特殊"命令构成. 这个长列表被写成一个.dvi文件
2. 其它一些程度读这个.dvi文件并把这些字母–坐标对翻译成PostScript命令,这些命令可以把给定的字母放在给点的坐处.

关于这个过程的一个经典的例子是latex结合dvips的使用.latex程序(就是一个预安装上LATEX 宏的tex程序)产生一个.dvi文件作为它的输出.dvips程序使用这个输出文件并产生一个.ps文件(PostScript文件).可能的话,这个文件还可以进一步的被其它程序使用,如ps2pdf,从其名字就可看出它的意思是"PostScript 到PDF".使用这个过程的另一个例子是tex 结合dvipdfm使用.dvipdfm程序把一个.dvi文件作为其输入并把其中的字母–坐标对转化为PDF 命令,直接得到一个.pdf文件.最后,tex4ht也是一个以.dvi文件为其输入产生输出的程序,只不过这次它的输出是一个.html文件.程序pdftex 和pdflatex有些特别:它们直接产生一个.pdf文件而跳过了中间过程.dvi阶段.但是从程序员的角度来看,它们执行过程中好像还是有那个是中间过程.

一般地,TEX 只产生字母–坐标对作为它的"输出".这明显的很难用于画图,如一条曲线.为此,要使用一些"特殊的"命令.不幸的是, 不同的产生.dvi文件的程序,它们的特殊命令是不一样的.事实上,每一个以.dvi文件作为其输入的程序,其特殊命令的语法表达式,完全不同.

PGF 的主要任务之一就是要"抽象"出不同程序中语法的不同.但是,这同时意味着对每一个程序的支持都必须要"编程" ,而这是一个费时而复杂的工作.

7.2.1 选择后台驱动

当TEX 排版你的文档,它并不知道你要用什么程序来把文档转化成.dvi文件.如果你的文件不包含任何特殊命令,那还算好;但是这年头几乎所有的.dvi文件都包含大量的特殊命令.因此,告诉TEX 你准备使用的是哪一个程序是非常有必要的.

不幸的是,没有"标准"的方法来告诉TEX 这件事.对于LATEX 格式来说,在其graphics宏包中有一个极其复杂的机制,PGF 就嵌入于这种机制之中.对于其它格式,当这种嵌入并不人所愿时,你有就必要直接告诉PGF 你准备使用哪一个程序.要完成这件事在使用pgf之前,你可以重新定义宏\pgfsysdriver并赋予其适当的值.如果你准备使用dvips 程序,你把这个宏的值赋以pgfsys-dvips.def; 如果你使用pdftex 或pdflatex,你赋以pgfsys-pdftex.def;等等.下面,我们来讨论对不同的程序的支持的相关细节.

7.2.2 产生 PDF 输出

PGF 支持三个程序来产生PDF 输出(PDF 意思是"可移植文本格式",为Adobe 公司发明):dvipdfm, pdftex, 和vtex. pdflatex程序与pdftex程序是一样的:它们使用不同的输入格式,但是输出格式却一模一样.

File pgfsys-pdftex.def

这是使用pdfTEX 的驱动文件,也就是说,它带有pdftex 或pdflatex命令.它引用pgfsys-common-pdf.def.

这个驱动有"完全"的功能.也就是说,任何PGF "完全可以做到"的工作,用它都可以实现.

File pgfsys-dvipdfm.def

这是使用(la)tex加dvipdfm的驱动文件.它引用pgfsys-common-pdf.def.

这个驱动支持大部分PGF 的功能,但是也有一些限制:

1. LaTeX 模式中,它使用graphicx来插入图片,而且不支持遮盖.
2. 在plain TeX 中,它不支持插入图片
3. 对remembering of pictures[10] (inter-picture connections),你需要使用pdftex的最近版本并以DVI-mode运行.

File pgfsys-vtex.def

这是使用商业VTEX程序的驱动文件.尽管它产生PDF输出,但是它仍然引用pgfsys-common-postscript.def文件. 注意到VTEX程序可以*同时*产生Postscript 和PDF输出,这依赖于它的命令行参数.但是,不管你是要产生Postscript 还是PDF输出,不要对这个驱动做任何的改动.

这个驱动支持大部分的PGF 的功能,除了以下一些限制:

1. LaTeX 模式中,它使用graphicx来插入图片,而且不支持遮盖.
2. 在plain TeX 中,它不支持插入图片
3. 阴影效果可完全实现,但输出效果与dvips的实现相同.
4. 不支持不透明性
5. 不支持remembering of pictures (inter-picture connections)

当然也可以通过以下方法来产生一个.pdf文件:先产生一个PostScript文件(如下)然后使用一个PostScript-to-PDF 转换程序,如ps2pdf 或the Acrobat Distiller.

7.2.3 产生PostScript输出

File pgfsys-dvips.def

This is a driver file for use with (la)tex followed by dvips. It includes pgfsys-common-postscript.def.

This driver also supports most of PGF's features, except for the following restrictions: 这是使用(la)tex加dvips的驱动文件,它引用pgfsys-common-postscript.def.

这个驱动也支持大部分PGF 的功能,除了以下一些限制:

1. LaTeX 模式中,它使用graphicx来插入图片,而且不支持遮盖.
2. 在plain TeX 中,它不支持插入图片
3. 阴影效果可完全实现,但输出效果不如采用某个驱动而生的.pdf
4. 不透明性只有在与新版本的GhostScript同在的时候才会工作.
5. 对remembering of pictures (inter-picture connections),你需要使用pdftex的最近版本并以DVI-mode运行.

File pgfsys-textures.def

这是使用TEXTURES程序的驱动文件.它引用pgfsys-common-postscript.def.

这个驱动与dvips驱动具有完全相同的限制.

你也可以使用vtex程序配合pgfsys-vtex.def来生成Postscript输出.

[10] 不知如何译

7.2.4 产生HTML / SVG 输出

tex4ht程序把.dvi文件转化为.html文件.HTML格式不能绘图,而SVG格式可以. 使用以下驱动,你可以要求PGF 为你文本中的每个PGF 图像产生一个SVG-图片

File pgfsys-tex4ht.def

这是使用tex4ht程序的驱动文件.它引用pgfsys-common-svg.def.

当使用这个驱动时,你应该注意以下的限制:

1. LaTeX 模式中,它使用graphicx来插入图片,而且不支持遮盖.
2. 在plain TeX 中,它不支持插入图片
3. 不支持remembering of pictures (inter-picture connections)
4. 对pgfpicture图中的文本支持不够好.原因是SVG规范至今对文本支持都不好,而且也不可能正确的"溜回"HTML. 所有的这些问题将来可能会解决,但是现今只有两种文本正常显示:第一,不带有数学模式,特殊符号或其它一些特别东西的简单文本. 第二,*非常*简单的只包含上标和下标的数学文本.尽管如此,变量还是不能正常的显示为斜体,而且通常情况下,文字看上去也很丑.
5. 如果你使用了包含任何特殊的东西,哪怕简单如α,都可能毁掉图形,因为text4ht并不总能正常的生成有效的XML代码. 因此,再说一遍,*在图形中坚持使用非常简单的结点*,不好意思.
6. 不像其它的输出格式,这里图形的外周盒子,"确实会修剪"图片.

驱动基本上是如下工作:当一个{pgfpicture}开始后,会调用适当的\special 命令来指导tex4ht,并输出到一个叫做\jobname-xxx.svg 的文件,这里xxx是一个数字,它会随着图片的增加而增加.然后,直到图形的结束,每一个(系统层)的图形命令都建立一个特别命令, 它们把适当的SVG字面文字插入到输出文件中.精确的细节有些复杂,因为PostScript/PDF 与SVG 的绘图模型和处理模型不完全相同;但是它们已经"足够接近",PGF可以达到其目的了.

7.2.5 产生完美的可移植DVI输出

File pgfsys-dvi.def

This is a driver file that can be used with any output driver, except for tex4ht.

The driver will produce perfectly portable .dvi files by composing all pictures entirely of black rectangles, the basic and only graphic shape supported by the TeX core. Even straight, but slanted lines are tricky to get right in this model (they need to be composed of lots of little squares).

Naturally, *very little* is possible with this driver. In fact, so little is possible that it is easier to list what is possible:

这个驱动文件可以供tex4ht之外的任何驱动使用.

这个驱动可以产生完美的可移植.dvi文件,它完全用TeX 核心支持的唯一基本的外形–黑矩形来构成图形, 甚至包括直线.但是斜线在这种模式下要正确显示需要点小窍门(它们需要由大量的小正方形组成).

自然的,这个驱动的功能非常的少.事实上,功能少到我们可以只列出它能做什么:

- 文本盒子可以以通常的方式放置.
- 直线和曲线可以画上(删掉).如果它们不是水平或竖直的,它们可由成百上千个小矩形组合而成.
- 支持不同粗细的线
- 支持坐标变换

注意一下,甚至连填充也不支持!(更不要说颜色或其它精美的东西了.)

这个驱动只有一个真正的用途:当你在图中只需要水平或是竖直的线时,这就派上用场了,结果相当令人满意.