1 Introduction, Basics

The PSTricks package has some very nice tools for drawing trees. Here is some minimal documentation.

You can get PostScript documentation, and pdf.

For fuller documentation, downloading, etc. see:

- The PSTricks chapter of The LaTeX Graphics Companion, by Michel Goossens, Sebastian Rahtz and Frank Mittelbach, Addison Wesley, 1997

Beware, this package relies heavily on PostScript ‘specials’ in the output. This means it only works properly with PostScript. Hence (a) users of pdf beware, not everything may work; in particular, it will not work if you use pdflatex, however, using ps2pdf to produce the pdf seems to work okay. (b) Users of DVI beware, what you see under xdvi may not be what you will ultimately get — in particular, some things that are actually working will not look right under when viewed with xdvi, so always check the PostScript, e.g. with ghostview.

Usage: in the preamble, put:

```
\usepackage{pstricks,pst-node,pst-tree}
```

The command to make a tree is \pstree[settings]{RootNode}{DaughterNodes}, the Root node is formed with a command like \TR, the terminal daughters are formed the same way. Daughters can also be trees (formed with \pstree).

Here is a simple example:
Alternatively, you can use a \begin{psTree}...\end{psTree} environment.

The remainder of this document just consists of playing around with various parameters to achieve a variety of effects.

## 2 Simple Variations

Increase the gap between levels of the tree with \texttt{levelsep=N}; leave more/less space around nodes with \texttt{nodesep=N}:

Setting \texttt{nodesep=2pt} makes for a more pleasing gap between the node label and the branches:
Setting \texttt{levelsep=5ex} means there is only this much distance between the levels of the tree:

\begin{verbatim}
\pstree[\texttt{nodesep=2pt, levelsep=5ex}]{{\TR{Node} }}{
\TR{D1}
\pstree{ {\TR{D2} }}{
\TR{D2.1}
\TR{D2.2}
}\TR{D3}
\TR{D4}
}\end{verbatim}

You can put labels on branches by adding \texttt{\ncput, \tlput, or \trput} commands in appropriate places (after the \texttt{\TR}, for example). Of course, this may result in text going over the lines – you may have to increase the separation between subtrees to fix this (I have not done this here):

\begin{verbatim}
\pstree[\texttt{\texttt{\texttt{levelsep=5ex, nodesep=2pt}}} ]{{\TR{Node} }}{
\TR{D1}\tlput{head}
\pstree{ \TR{D2}\trput{subj} }{
\TR{D2.1}\tlput{head}
\TR{D2.2}\trput{dep}
\TR{D2.3}\trput{dep2}
}\TR{D3}\trput{obj}
}\end{verbatim}

Put nodes in circles with \texttt{\Tcircle{Node}}, instead of \texttt{\TR}:

\begin{verbatim}
\pstree[\texttt{\texttt{\texttt{levelsep=8ex}}} ]{{\Tcircle{R} }}{
\Tcircle{D1}
\pstree{ \Tcircle{D2} }{
\Tcircle{D2.1}
\Tcircle{D2.2}
}\Tcircle{D3}
\Tcircle{D4}
}\end{verbatim}
In the same way, you can get other shapes around nodes: diamonds (\Tdia{Node}), ovals (\Toval{Node}), triangles (\Ttri{Node}).

You can change the way branches are drawn in many ways, e.g. dotted lines:

\pstree{\Tcircle{R}}{
  \Tcircle{D1}
  \pstree{\Tcircle{D2}}{
    \psset{linestyle=dashed}
    \Tcircle{D2.1}
    \Tcircle{D2.2}
  }
  \Tcircle{D3}
  \Tcircle{D4}
}

(The command \psset, which sets this parameter, can appear anywhere, it has the scope you would expect).

Here are some other kinds of node. This also shows how to get a subtree drawn under a triangle: draw a tree whose root is a triangle, and whose daughter is the material to go underneath; make the linestyle of this subtree none, put no ends on arrows, and make the level separation small. You have to make sure that the triangle is big enough, with the optional argument to \Tfan.
Here we put labels on arcs, and make the arcs themselves downwards pointing arrows, rather than lines, with `arrows=->`:

\pstree[levelsep=20ex]{\Toval{R}} {
\Tdia{D1}
\pstree{ \Tcircle{D2}\ncput{subj} } {
\TR{D2.1}\tlput{head}
\Tr{D2.2}\trput{dep}
\Tr{D2.3}\trput{dep2}
}
\Toval{D3}
\pstree[linestyle=none, arrows=-, levelsep=1ex]{\Tfan[fansize=20ex]}{\TR{material}}
\Ttri{D4}
}\psset{arrows=->}
\pstree[levelsep=10ex]{\Tcircle{R}} {
\Tcircle{D1}
\pstree{ \Tcircle{D2}\tvp{subj} } {
\Tcircle{D2.1}\tlput{head}
\Tcircle{D2.2}\trput{dep}
}
\Tcircle{D3}
\Tcircle{D4}
}\psset{arrows=->}
\pstree[levelsep=10ex]{\Tcircle{R}} {
\Tcircle{D1}
\pstree{ \Tcircle{D2}\tvp{subj} } {
\Tcircle{D2.1}\tlput{head}
\Tcircle{D2.2}\trput{dep}
}
\Tcircle{D3}
\Tcircle{D4}
}

Notice the commands that label the branches come after the node at the end of the branch. The `\ncput`, `\tlput`, and `\trput` differ in where they put the label in relation to the branch:

- `\tvp{something}` puts something in the middle of the branch (based on vertical distance between nodes)
- `\tap{something}` puts something above the branch
- `\bp{something}` puts something below the branch
- `\hvp{something}` puts something in middle of the branch (based on horizontal distance between nodes)
- `\rvp{something}` puts something right of the branch
- `\lvp{something}` puts something left of the branch
Turn the tree upside down (make it grow from bottom to top), with \texttt{treemode=U}: Notice that the labels are the same side of the line as before.

\begin{verbatim}
\psset{arrows=->}
\pstree[treemode=U,levelsep=10ex]{\Tcircle{R} }{
  \Tcircle{D1}
  \pstree{ \Tcircle{D2}\vput{subj} }{
    \Tcircle{D2.1}\lput{head}
    \Tcircle{D2.2}\rput{dep}
  }
  \Tcircle{D3}
  \Tcircle{D4}
}
\end{verbatim}

Turn the whole thing on its side with \texttt{treemode=R} (notice that we have changed \texttt{\lput} to \texttt{\rput}, etc.) to fit with the orientation of the tree:

\begin{verbatim}
\pstree[treemode=R,levelsep=12ex]{\Tcircle{R} }{
  \Tcircle{D1}
  \pstree{ \Tcircle{D2}\rput{subj} }{
    \Tcircle{D2.1}\tput{head}
    \Tcircle{D2.2}\tput{dep}
  }
  \Tcircle{D3}
  \Tcircle{D4}
}
\end{verbatim}

To use curves instead of straight lines, redefine \texttt{\psedge}: Here it is redefined, and the angle at which branches go into the daughter nodes is also adjusted, as is the separation between levels.
3 Non-local Tree Lines

Nodes can be given ‘names’, and lines drawn between them. One of the drawing commands is \texttt{ncurve\{settings\}\{arrows\}} which draws a Bezier curve between the nodes from and to. The sharpness of the curve is controlled by the \texttt{ncurv} parameter. Higher values give sharper curves. \texttt{ncurvA} controls the first part of the curve, \texttt{ncurvB} the second part.

The parameter \texttt{angleA} says what angle to leave the ‘from’ node (nothing means leave from the right; 90 means leave from the top, 180 from the left, so increased values take you further counter-clockwise). Parameters \texttt{angleB} does the same for the to node. The direction of the curve (hence the effect of increasing the value of \texttt{ncurv}) depends on this.

In this example, the line from d3 has to leave from below, and arrive from below in order to clear the rest of the tree, it has to do this with a fairly high curve value.

This gets to be quite intuitive after a while...

Since these lines are drawn with PostScript, \LaTeX{} does not know exactly how much space to leave for them. In the last example, one of the curves goes well outside where \LaTeX{} thinks the material ends. To avoid it going over surrounding text, I have put it in a figure, and added a 5ex high ‘strut’ \texttt{\rule{0pt}{5ex}} to make \LaTeX{} think the figure is bigger than it actually is.