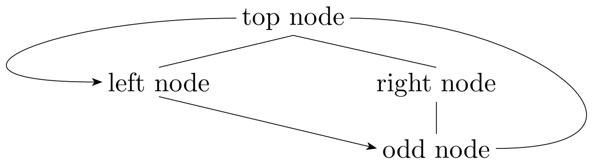
Tree Macros

Emma Pease

May 10, 1995

The tree macros package allows one to integrate LaTeX and Postscript. For example, one can use LaTeX to layout a tree and have Postscript draw the lines.



These macros work by defining locations on a page and then manipulating them in a variety of ways. The commands that created the above tree are as follows:

```
\begin{tabular}{ccc}
                    &\node{a}{top node}\\[3ex]
\node{b}{left node} &     & \node{c}{right node}\\[3ex]
                    &     & \node{d}{odd node}
\end{tabular}
\nodeconnect{a}{b}
\nodeconnect{a}{c}
\nodeconnect{c}{d}
\nodecurve[r]{a}[r]{d}{1in}
\anodeconnect[b]{b}[l]{d}
\anodecurve[l]{a}[l]{b}{1in}
```

You will notice that four nodes are defined, a, b, c, and d, using the `\node` command. These nodes are then connected using the `\nodeconnect` and `\nodecurve` commands.

1 Locating Commands

Location commands are those that deal with defining a location on a page. The basic command

```
\node{nodename}{object}
```

Each node has its name, height, width, and the location of the lower left hand corner point passed down to postscript where it will remain until needed. Note that the object will be printed by TEX but the lines drawn by Postscript. A variant of this command is

```
\nodepoint{nodename}[horizontal displace][vertical displace]
```

The node's height and width are 0pts, but the location can be displaced.

2 Connecting Commands

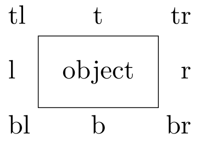
These commands connect two or more nodes.

2.1 nodeconnect

One basic command is

```
\nodeconnect[fromloc]{fromnodename}[toloc]{tonodename}
```

fromnodename and tonodename must be the names of two existing nodes. Imagine the node as a box, fromloc and toloc are the locations on that box to draw the connecting lines.

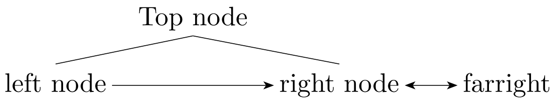


The present choices are t [top], b [bottom], l [left], r [right], tl [topleft], tr [topright], bl [bottomleft], and br [bottomright]. These could be expanded. By default the fromloc is bottom and the toloc is top.

Variants of this command are

```
\anodeconnect[fromloc]{fromnodename}[toloc]{tonodename}
\aanodeconnect[fromloc]{fromnodename}[toloc]{tonodename}
```

The first places an arrow pointing to the second node. The second puts arrows on both ends of the line.



```
\begin{tabular}{ccccc}
                    &\node{a}{Top node}\\[3ex]
\node{b}{left node\strut} &   & \node{c}{right node\strut}
&\qquad &\node{d}{farright\strut}
\end{tabular}
\nodeconnect{a}{b}
\nodeconnect[t]{c}[b]{a}
\anodeconnect[r]{b}[l]{c}
\aanodeconnect[r]{c}[l]{d}
```

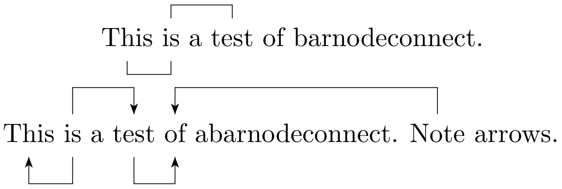
Note the `\strut`. This ensures that both nodes have the same height and depth.

2.2 barnodeconnect

Another way of connecting is via the bar connect commands

```
\barnodeconnect[depth]{fromnodename}{tonodename}
\abarnodeconnect[depth]{fromnodename}{tonodename}
```

For example,


This is a test of barnodeconnect.
This is a test of abarnodeconnect. Note arrows.

```
\node{c}{\strut This} \node{a}{\strut is} a \node{b}{\strut test}
of barnodeconnect.
\barnodeconnect{a}{b}
\barnodeconnect[-5pt]{a}{c}
\bigskip\bigskip

\node{c}{\strut This} \node{a}{\strut is} a \node{b}{\strut test}
\node{d}{\strut of} abarnodeconnect.  \node{e}{\strut Note} arrows.
\abarnodeconnect[10pt]{a}{b}
\abarnodeconnect[-10pt]{a}{c}
\abarnodeconnect[-10pt]{b}{d}
\abarnodeconnect[10pt]{e}{d}
```

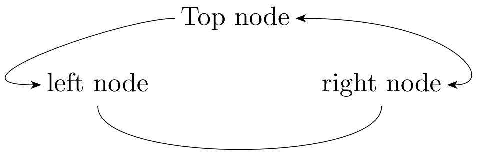
A negative depth places the bar below the line; a positive depth (or the default, which is 5pt) places the bar above the line.

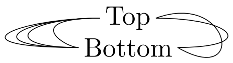
2.3 nodecurve

The nodecurve commands allow curves between nodes.

```
\nodecurve[fromloc][fromang]{fromnodename}[toloc][toang]{tonodename}{fdepth}[tdepth]
\anodecurve[fromloc][fromang]{fromnodename}[toloc][toang]{tonodename}{fdepth}[tdepth]
\aanodecurve[fromloc][fromang]{fromnodename}[toloc][toang]{tonodename}{fdepth}[tdepth]
```

The options fromloc and toloc are the same as for `\nodeconnect`. The options fromang and toang are the angle of incidence in degrees to the location with 0 being perpendicular and the default. Angles are calculated counterclockwise. fdepth and tdepth are dimensions and allow one to adjust how curved the curve is.





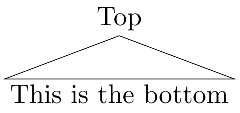
```
\begin{tabular}{ccc}
                    &\node{a}{Top node}\\[3ex]
\node{b}{left node\strut} &                 & \node{c}{right node\strut}
\end{tabular}
\nodecurve[b]{b}[b]{c}{.3in}
\anodecurve[l]{a}[l]{b}{20pt}[40pt]
\aanodecurve[r]{a}[r]{c}{60pt}[20pt]
\hfill
\begin{tabular}{c}
\node{d}{Top}\\
\node{e}{Bottom}
\end{tabular}
% here follow some curves that adjust fdepth and tdepth
\nodecurve[l]{d}[l]{e}{30pt}
\nodecurve[l]{d}[l]{e}{30pt}[40pt]
\nodecurve[l]{d}[l]{e}{30pt}[50pt]
\nodecurve[l]{d}[l]{e}{30pt}[5pt]

% here follow some curves that adjust toang and fromang
\nodecurve[r]{d}[r]{e}{30pt}
\nodecurve[r][20]{d}[r][-30]{e}{30pt}
```

2.4 Other connecting commands

A few odd commands

```
\nodetriangle{fromnodename}{tonodename}
```

This creates a triangle whose apex is the bottom of `fromnodename` and whose base is the top of `tonodename`.

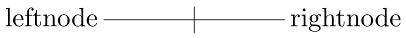


```
\begin{tabular}{c}
\node{a}{Top}\\[4ex]
\node{b}{This is the bottom}
\end{tabular}
\nodetriangle{a}{b}
```

The last command is meant to be used with the `\nodeconnect` command. It causes a short line to cross perpendicular to the line.

```
\delink[fromloc]{fromnodename}[toloc]{tonodename}{length}
```

An example follows

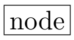


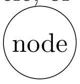
3 Single Node commands

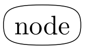
These commands adjust something around a single node rather than connecting nodes. The basic commands are

```
\nodebox{nodename}
\nodecircle[depth]{nodename}
\nodeoval{nodename}
```

They draw, respectively, a box, circle, or oval around the given node.







You will probably wish to call these commands after you have called all the connecting commands you will be using in a particular diagram.

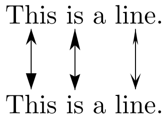
4 Parameters

At the moment there are three parameters that can be changed. They are

- `\nodemargin` - A node's height and width are defined as the height plus depth and width of an hbox enclosing the object plus the nodemargin on each side. The default is 2pt.
- `\treelinewidth` - The width of the lines. The default is .3pt.

- `\dashlength` - The length of the dash, if you are using dashed lines. The default is 0pt (solid line).[1]
- `\arrowwidth` - the width of the arrowhead in the `\anodeconnect` and `\anodecurve` commands. Default is 3 pt.
- `\arrowlength` - the length of the arrowhead. Default is 4pt.
- `\arrowinset` - the inset in the arrow. Default is 1pt.

The command `\arrowhead{width}{length}{inset}` allows one to define all three parameters in one go.



```
\node{a}{This} \node{b}{is} a \node{c}{line}.\\[5ex]
\node{d}{This} \node{e}{is} a \node{f}{line}.%
% The arrows pointing down give examples of differing insets of arrowheads
\arrowhead{4pt}{6pt}{0pt}%
\anodeconnect{a}{d}%
\arrowhead{4pt}{6pt}{1pt}%
\anodeconnect{b}{e}%
\arrowhead{4pt}{6pt}{3pt}%
\anodeconnect{c}{f}%

% the arrows pointing up give examples of differing lengths
% and widths of the arrowheads.
\arrowhead{4pt}{6pt}{2pt}%
\anodeconnect[t]{d}[b]{a}%
\arrowhead{4pt}{8pt}{2pt}%
\anodeconnect[t]{e}[b]{b}%
\arrowhead{2pt}{6pt}{2pt}%
\anodeconnect[t]{f}[b]{c}%
```

5 How to Run

Add the style file, tree-dvips.sty,

```
\documentstyle[tree-dvips]{article}
```

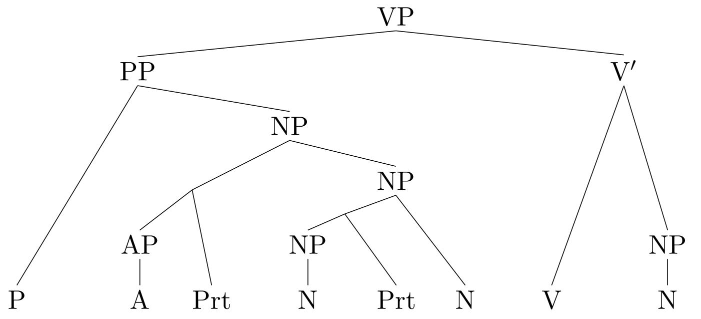
Run through LaTeX and send to a postscript printer using dvips (written by Tomas Rokicki).

---

[1] The length of the dash and the length between the dashes are the same. An exercise for someone who knows postscript and tex is to allow the dash and the blank to vary in size.

6 Examples

A series of examples follow.



‘zaw	‘oN	‘geq	?njiaw	‘geq	‘dou	khe	tshjaN
toward	red		bird		head	open	gun

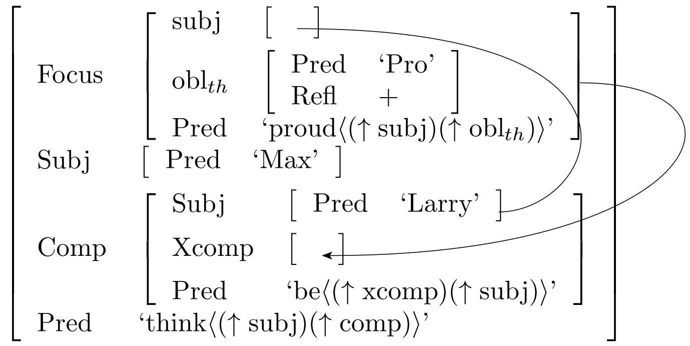
‘shoot at the red head of the bird’

```
\let\mc=\multicolumn
\begin{tabular}[t]{@{}lllllllll@{}}
    &    &    &      &\node{a}{VP} \\[2ex]
    &\node{b}{PP}
         &    &      &    &    &\mc{2}{c}{\node{c}{V$’$}}\\[2ex]
    &    &    &\node{d}{NP} \\[2ex]
    &    &\nodepoint{e}
              &      &\node{f}{NP} \\
    &    &    &\hfill\nodepoint{r}[0pt][3pt]   \\
    &\hfil\node{g}{AP}
         &    &\hfil\node{h}{NP}
                     &    &    &     &\hfil\node{i}{NP}\\[2ex]
\node{j}{P}
    &\hfil\node{k}{A}
         & \node{l}{Prt}
              &\hfil\node{m}{N}
                     &\node{n}{Prt}
                          &\node{o}{N}
                               &\node{p}{V}
                                     &\hfil\node{q}{N}\\[2ex]
‘zaw&‘oN &‘geq&?njiaw&‘geq&‘dou& khe &tshjaN \\
toward &red &    & bird &    &head& open&gun \\[1ex]
\mc{9}{@{}l}{‘shoot at the red head of the bird’} \\
\end{tabular}
\nodeconnect{a}{b}     \nodeconnect{e}{l}
\nodeconnect{a}{c}    \nodeconnect{d}{f}
```

```
\nodeconnect{b}{d}     \nodeconnect{g}{k}
\nodeconnect{b}{j}     \nodeconnect{f}{r}
\nodeconnect{c}{p}     \nodeconnect{r}{h}
\nodeconnect{c}{i}     \nodeconnect{r}{n}
\nodeconnect{d}{e}     \nodeconnect{f}{o}
\nodeconnect{e}{g}     \nodeconnect{h}{m}
\nodeconnect{i}{q}
```

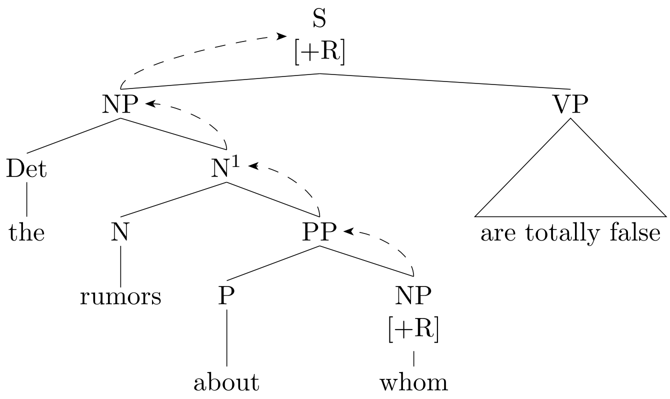
The following two examples use `\outerfs` and `\modsmalltree`; these are both part of the lingmacros package. See `lingmacros.sty` for more information.

(100)



```
\enumsentence[(100)]{\evnup[2pt]
{\outerfs{
Focus & \outerfs{subj       & \outerfs{\ \nodepoint{a}\ }\\[1ex]
                 obl$_{th}$ & \outerfs{Pred & ‘Pro’\\
                                       Refl & +}\\[1ex]
                 Pred       & ‘proud$\langle(\uparrow {\rm subj})
                             (\uparrow {\rm obl}_{th})\rangle$’}%
\nodepoint{d}[-3pt][0pt]\\[2ex]
Subj & \outerfs{Pred & ‘Max’}\\[1ex]
Comp & \outerfs{Subj &\outerfs{Pred & ‘Larry’}%
\nodepoint{c}[-3pt][0pt]\\[1ex]
                Xcomp&\outerfs{\ \nodepoint{b}\ }\\[1ex]
                Pred & ‘be$\langle(\uparrow {\rm xcomp})
                       (\uparrow {\rm subj})\rangle$’}\\[1ex]
Pred & ‘think$\langle(\uparrow {\rm subj})
                     (\uparrow {\rm comp})\rangle$’}\\
}
\nodecurve[r]{a}[r]{c}{2in}[.5in]
\anodecurve[r]{d}[r]{b}{1in}[2in]
}
```

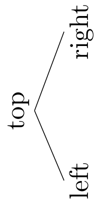
(1)



```
\enumsentence{\modsmalltree{6}{
     &       &   &\ns\node{a}{\begin{tabular}[t]{@{}c@{}}S\\
                        {}[+R]\end{tabular}}   \\
     &\node{b}{NP}
             &   &  && \node{c}{VP} \\
\node{d}{Det}
     &       &\node{e}{N$^1$} \\
\node{h}{the}
     &\node{i}{N}
             &   &\node{f}{PP}
                    &&\node{g}{are totally false}\\
     &\node{n}{rumors}
             &\node{j}{P}
                 &  &\ns\node{k}{\begin{tabular}[t]{@{}c@{}}NP\\
                                 {}[+R] \end{tabular}}\\
     &       &\node{l}{about}
                 &  &\node{m}{whom}}
\nodeconnect{a}{b}         \nodeconnect{e}{f}
\nodeconnect{a}{c}         \nodeconnect{i}{n}
\nodeconnect{b}{d}         \nodeconnect{f}{j}
\nodeconnect{b}{e}         \nodeconnect{f}{k}
\nodeconnect{d}{h}         \nodeconnect{j}{l}
\nodeconnect{e}{i}         \nodeconnect{k}{m}
\nodetriangle{c}{g}
{\makedash{4pt}
\anodecurve[t]{b}[l]{a}{10pt}  \anodecurve[t]{f}[r]{e}{10pt}
\anodecurve[t]{e}[r]{b}{10pt}  \anodecurve[t]{k}[r]{f}{10pt}
}}
```

7 Rotation

With the `rotate.sty` file one can also rotate figures. This is useful with wide figures that won't fit on within the page boundaries unless turned sideways. Earlier versions of the tree macros didn't work with this.



```
\rotate[l]{\modsmalltree{3}{&\node{a}{top}\\
\node{b}{left} && \node{c}{right}}
\nodeconnect{a}{b}
\nodeconnect{a}{c}
}
```

Note that the connection commands are within the boundaries of the `\rotate` command.

8 Errors

A multitude of caveats.

- Any commands calling nodes must be read while TEX is still processing the page the nodes are defined on. In other words don't define the nodes on page 1 and connect them with commands that appear at the end of the paper.
- Nodes mentioned in node connecting commands must exist or else the job won't print.
- Make sure the dvips postscript output is sent to a postscript printer. It is possible to send the dvi, but not the postscript, output to another printer; the lines just won't appear, assuming the printer ignores specials it doesn't know about.

9 Thanks

Thanks to all those who've helped me over the years, including Avery Andrews who revamped the arrows for me and Gintas Grigelionis who suggested how to finally get the rotation to work correctly.