Documentation of `mptrees.mp`

Olivier PÉAULT*

November 2, 2021

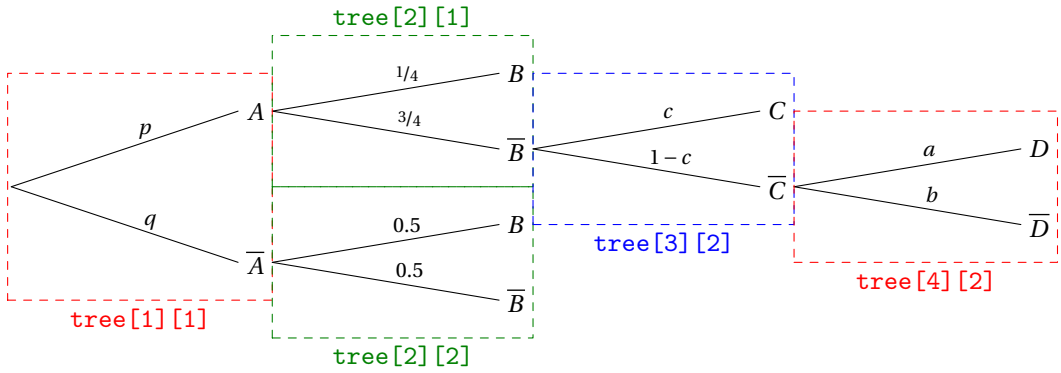
Contents

1 Overview

This package is intended to simplify the drawing of probability trees with METAPOST. It provides one main command and several parameters to control the output.

It can be used in standalone files with two compilations (`latexmp` package is loaded) but it can also be used with LuaLaTeX and `luamplib` package.

`tree[i][j](dim1,dim2,...)(ev1,prob1,ev2,prob2,...)` probability tree located in column `i` and row `j` (see figure below). `dim1`, `dim2`,... can be numerics or pairs and control the dimension of the tree. `ev1`, `prob1`... can be strings or pictures and will be printed (using `latexmp` if strings) at the end of the edge (the event) and above the edge (the probability).



*E-mail : o.peault@posteo.net

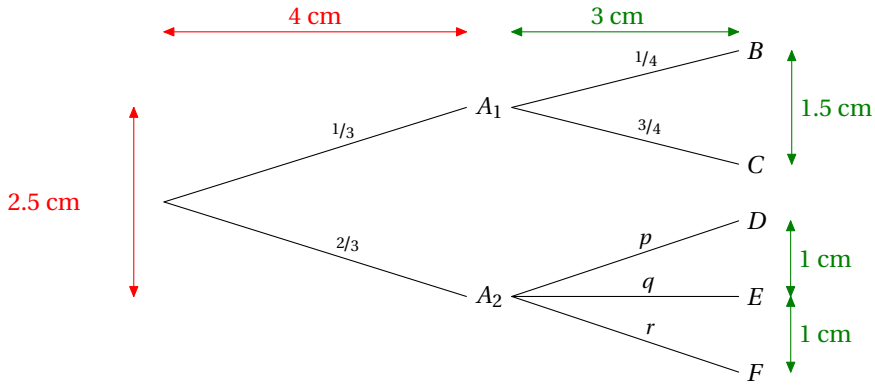
2 Trees

2.1 Different kinds of trees

`tree[i][j](width,vspace)(ev1,prob1,ev2,prob2,...)` regular tree where `width` is the horizontal width of the tree and `vspace` the vertical space between two consecutive nodes.

Example 1

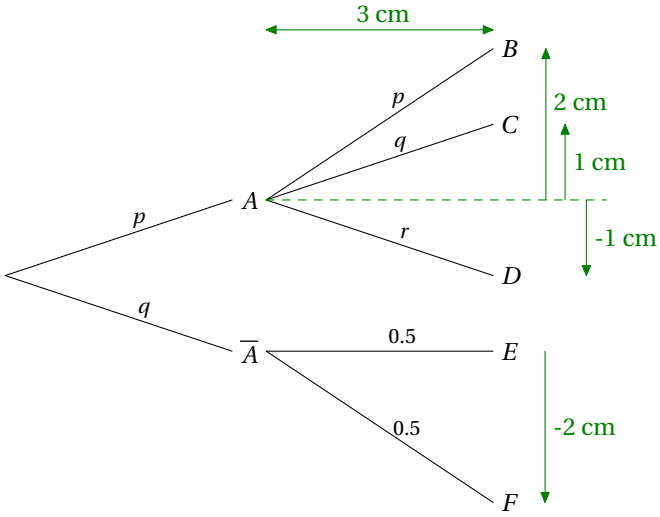
```
beginfig(1)
draw tree[1][1](4cm,2.5cm)("$A_1$","$\nicefrac{1}{3}$","$A_2$","$\nicefrac{2}{3}$");
draw tree[2][1](3cm,1.5cm)("$B$","$\nicefrac{1}{4}$","$C$","$\nicefrac{3}{4}$");
draw tree[2][2](3cm,1cm)("$D$","$p$","$E$","$q$","$F$","$r$");
endfig;
```



`tree[i][j](width,vspace1,vspace2...)(ev1,prob1,ev2,prob2,...)` tree where `width` is the horizontal width of the tree while each `vspace` indicates the vertical space between the node and the origin of the tree.

Example 2

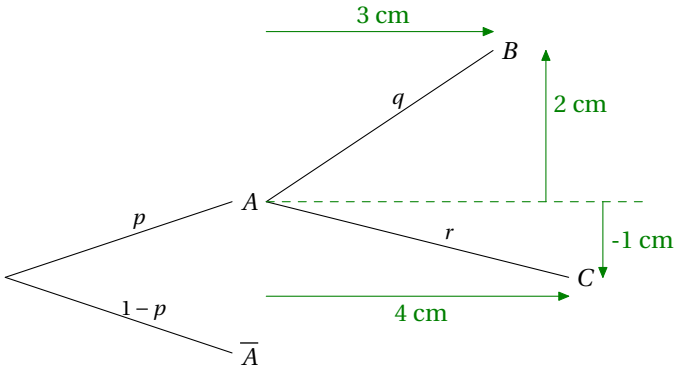
```
beginfig(2)
draw tree[1][1](3cm,2cm)("$A$","$p$","$\overline{A}$","$q$");
draw tree[2][1](3cm,2cm,1cm,-1cm)("$B$","$p$","$C$","$q$","$D$","$r$");
draw tree[2][2](3cm,0cm,-2cm)("$E$","$0.5$","$F$","$0.5$");
endfig;
```



`tree[i][j](pair1,pair2,...)(ev1,prob1,ev2,prob2,...)` tree where `pair1`, `pair2`... indicate the coordinates of each node from the origin of the tree.

Example 3

```
beginfig(3)
draw tree[1][1](3cm,2cm)("$A$","$p$","$\overline{A}$","$1-p$");
draw tree[2][1]((3cm,2cm),(4cm,-1cm))("$B$","$q$","$C$","$r$");
endfig;
```

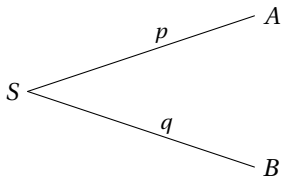


2.2 Start and end labels

`startlabel(s)` Print `s` (can be a string or a picture) at the origin of the tree.

Example 4

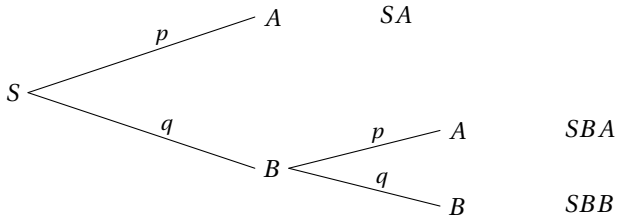
```
beginfig(4)
 draw startlabel("$S$");
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$");
endfig;
```



`endlabel[i][j](s)` Print `s` at the end of a branch. The space between the previous label ans `s` is controlled by the numeric `endlabelspace` which defaults to `1cm`.

Example 5

```
beginfig(5)
 draw startlabel("$S$");
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$");
 draw tree[2][2](2cm,1cm)("$A$","$p$","$B$","$q$");
 draw endlabel[2][1]("$SA$");
 draw endlabel[3][1]("$SBA$");
 draw endlabel[3][2]("$SBB$");
endfig;
```

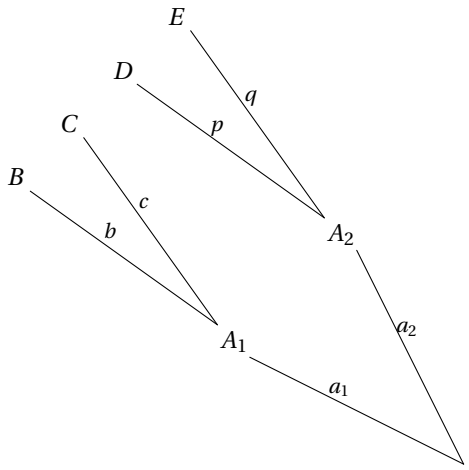


3 Direction

`dirtree` All trees are construct horizontally by default. `ditree` indicates the angle in degrees between the horizontal and the main direction of the tree. Default is 0.

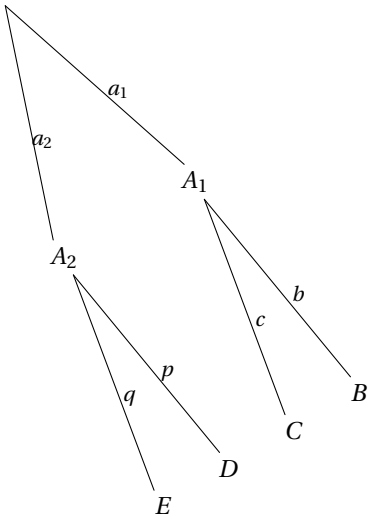
Example 6

```
beginfig(6)
dirtree:=135;
draw tree[1][1](3cm,2cm)("$A_1$","$a_1$","$A_2$","$a_2$");
draw tree[2][1](3cm,1cm)("$B$","$b$","$C$","$c$");
draw tree[2][2](3cm,1cm)("$D$","$p$","$E$","$q$");
endfig;
```



Example 7

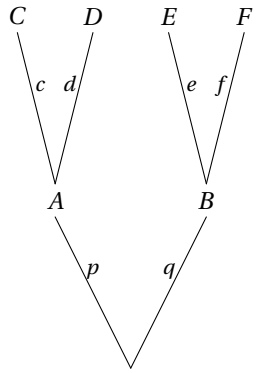
```
beginfig(7)
dirtree:=-60;
draw tree[1][1](3cm,2cm)("$A_1$","$a_1$","$A_2$","$a_2$");
draw tree[2][1](3cm,1cm)("$B$","$b$","$C$","$c$");
draw tree[2][2](3cm,1cm)("$D$","$p$","$E$","$q$");
endfig;
```



dirlabel With dirtree, the whole tree is rotated. With `dirlabel`, only the position of the labels is changed so the given coordinates are the real ones. May be useful for vertical trees.

Example 8

```
beginfig(8)
dirlabel:=90;
draw tree[1][1]((-1cm,2cm),(1cm,2cm))("$A$","$p$","$B$","$q$");
draw tree[2][1]((-0.5cm,2cm),(0.5cm,2cm))("$C$","$c$","$D$","$d$");
draw tree[2][2]((-0.5cm,2cm),(0.5cm,2cm))("$E$","$e$","$F$","$f$");
endfig;
```



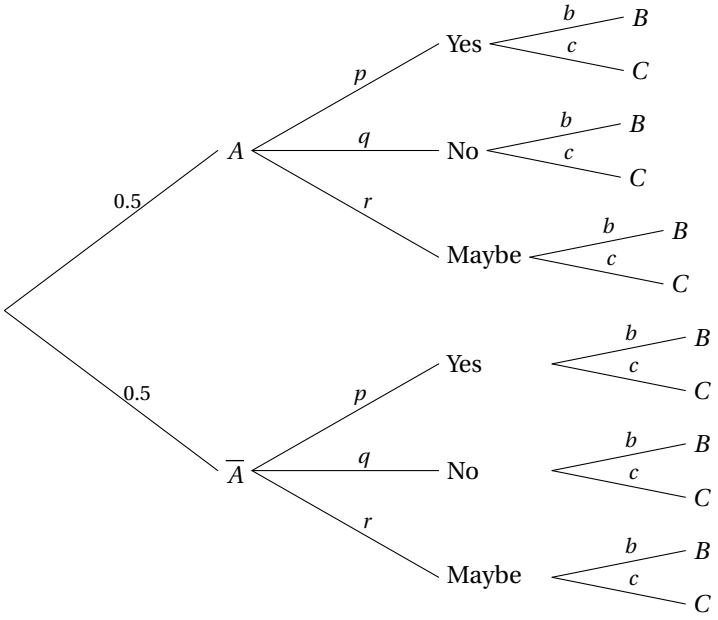
4 Dealing with alignment

shiftev The origin of each tree is located where the bounding box of the previous event's name ends. Thus subtrees may begin at different places. The numeric `shiftev` indicates the horizontal space between the end of the edges and the beginning of following trees.

It can be used inside the first set of parameters of the tree (see example below) or as a global variable.

Example 9

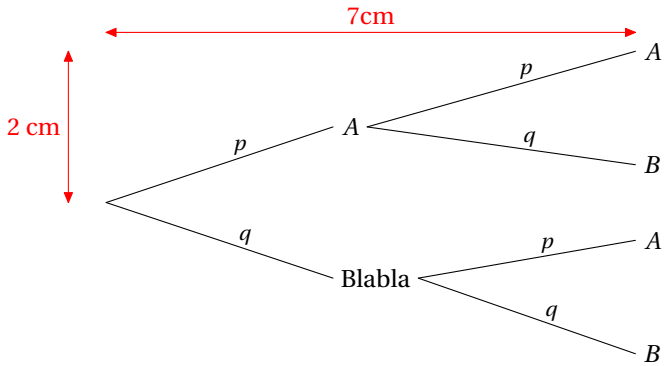
```
beginfig(9)
 draw tree[1][1](80,120)("$A$","$0.5$","$\overline{A}$","$0.5$");
 draw tree[2][1](70,40)("Yes","$p$","No","$q$","Maybe","$r$");
 draw tree[2][2](70,40,"shiftev:=1.5cm")("Yes","$p$","No","$q$","Maybe","$r$");
 draw tree[3][1](50,20)("$B$","$b$","$C$","$c$");
 draw tree[3][2](50,20)("$B$","$b$","$C$","$c$");
 draw tree[3][3](50,20)("$B$","$b$","$C$","$c$");
 draw tree[3][4](50,20)("$B$","$b$","$C$","$c$");
 draw tree[3][5](50,20)("$B$","$b$","$C$","$c$");
 draw tree[3][6](50,20)("$B$","$b$","$C$","$c$");
endfig;
```



abscoord With the boolean `abscoord` set to `true`, all the coordinates are given from the origin of the *first* tree instead of the origin of the subtree, which makes easier the alignment of all the subtrees.

Example 10

```
beginfig(10)
abscoord:=true;
draw tree[1][1](3cm,2cm)("$A$","$p$","Blabla","$q$");
draw tree[2][1]((7cm,2cm),(7cm,0.5cm))("$A$","$p$","$B$","$q$");
draw tree[2][2]((7cm,-0.5cm),(7cm,-2cm))("$A$","$p$","$B$","$q$");
endfig;
```



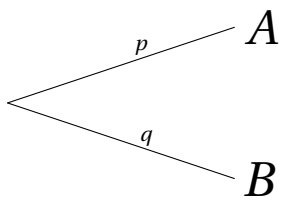
5 Parameters

5.1 Event

scaleev numeric controlling the scale of the label at the end of the edge (the event). Default is 1.

Example 11

```
beginfig(11)
 scaleev:=2;
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$");
endfig;
```

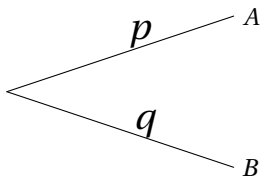


5.2 Probability

`scaleprob` numeric controlling the scale of the label above the edge (the probability). Default is 0.85.

Example 12

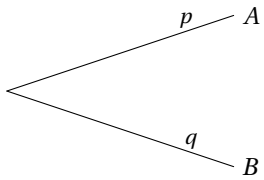
```
beginfig(12)
 scaleprob:=1.5;
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$");
endfig;
```



`posprob` numeric controlling the position of the label above the edge. Default is 0.6.

Example 13

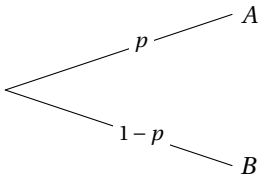
```
beginfig(13)
 posprob:=0.8;
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$");
endfig;
```



`typeprob` numeric controlling how the label is printed. Values can be 1 (the default, label is printed above the edge), 2 (the label is printed on the edge), 3 (the label is printed above the edge and rotated) or 4 (the label is printed on the edge and rotated).

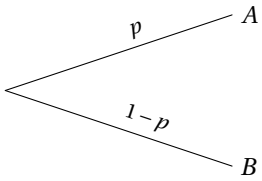
Example 14

```
beginfig(14)
 typeprob:=2;
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$1-p$");
endfig;
```



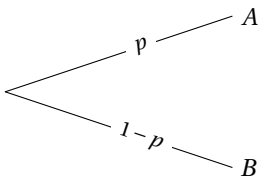
Example 15

```
beginfig(15)
 typeprob:=3;
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$1-p$");
endfig;
```



Example 16

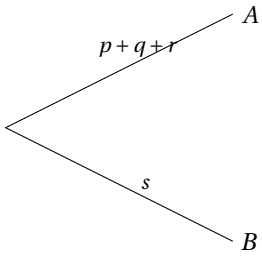
```
beginfig(16)
 typeprob:=4;
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$1-p$");
endfig;
```



proboffset numeric controlling the amount by which the label above the edge is offset. Default is `labeloffset` (3bp).

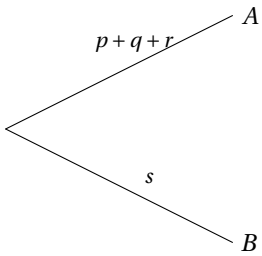
Example 17

```
beginfig(17)
 draw tree[1][1](3cm,3cm)("$A$","$p+q+r$","$B$","$s$");
endfig;
```



Example 18

```
beginfig(18)
 proboffset:=6bp;
 draw tree[1][1](3cm,3cm)("$A$","$p+q+r$","$B$","$s$");
endfig;
```

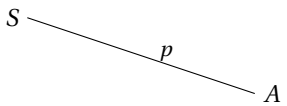


5.3 Edge

endedgeshift vertical space added at the end of the edge. Default is 0. Useful when various edges end at the same point

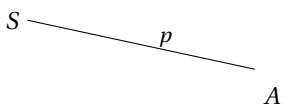
Example 19

```
beginfig(19)
 draw startlabel("$S$");
 draw tree[1][1]((3cm,-1cm))("$A$","$p$");
endfig;
```



Example 20

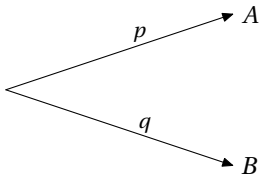
```
beginfig(20)
 endedgeshift:=10;
 draw startlabel("$S$");
 draw tree[1][1]((3cm,-1cm))("$A$","$p$");
endfig;
```



edgearrow When the boolean `edgearrow` is set to true, edges end with an arrow. Default is `false`.

Example 21

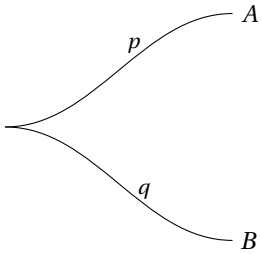
```
beginfig(21)
 edgearrow:=true;
 draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$");
endfig;
```



branchtype string which controls the shape of the edge. Possible values are `segment` (default), `curve`, `broken`. With the last one, the length of the first segment of the broken line depends on a numeric parameter, `brokenlineratio` which indicates the ratio between the length of this segment and the length of the horizontal space. Default value is `0.2`.

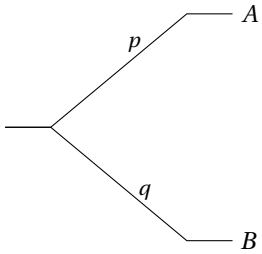
Example 22

```
beginfig(22)
 branchtype:="curve";
 draw tree[1][1](3cm,3cm)("$A$","$p$","$B$","$q$");
endfig;
```



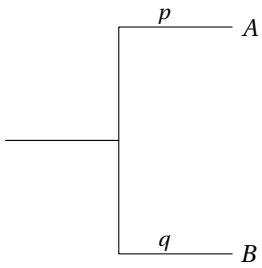
Example 23

```
beginfig(23)
 branchtype:="broken";
 draw tree[1][1](3cm,3cm)("$A$","$p$","$B$","$q$");
endfig;
```



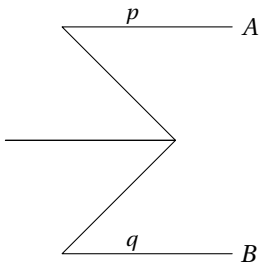
Example 24

```
beginfig(24)
 branchtype:="broken";
 posprob:=0.8;
 brokenlineratio:=0.5;
 draw tree[1][1](3cm,3cm)("$A$","$p$","$B$","$q$");
endfig;
```



Example 25

```
beginfig(25)
 branchtype:="broken";
 posprob:=0.8;
 brokenlineratio:=0.75;
 draw tree[1][1](3cm,3cm)("$A$","$p$","$B$","$q$");
endfig;
```



6 Binomial trees

bernoulliprocess(n)(l,h) tree describing the Bernoulli process with n trials. `l` is the length of the first edges and `h` is the space between two final nodes.

bernoulliprocessL(n)(L,H) same as above where `L` is the whole width of the tree and `H` its height.

Several parameters control the output:

bernoullisuccessevent string printed at every node representing a success. Default is `"$S$"`.

bernoullifailureevent string printed at every node representing a failure. Default is `"$\overline{S}$"`.

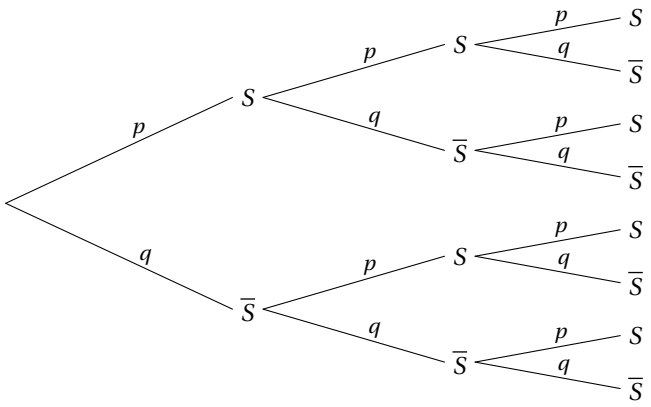
bernoullisuccessprob string printed above every edge representing a success. Default is `"$p$"`.

bernoullifailureprob string printed above every edge representing a success. Default is `"$q$"`.

bernoulliscalebranch ratio between width of consecutive edges. Default is `0.8`.

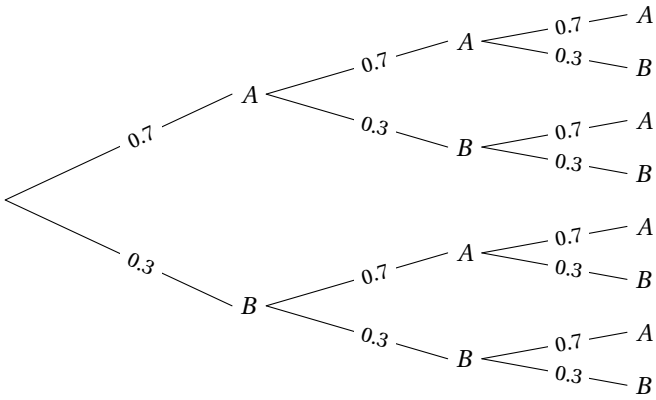
Example 26

```
beginfig(26)
draw bernoulliprocess(3)(3cm,0.7cm);
endfig;
```



Example 27

```
beginfig(27)
typeprob:=4;
bernoullisuccessevent:="$A$";
bernoullifailureevent:="$B$";
bernoullisuccessprob:="$0.7$";
bernoullifailureprob:="$0.3$";
draw bernoulliprocess(3)(3cm,0.7cm);
endfig;
```

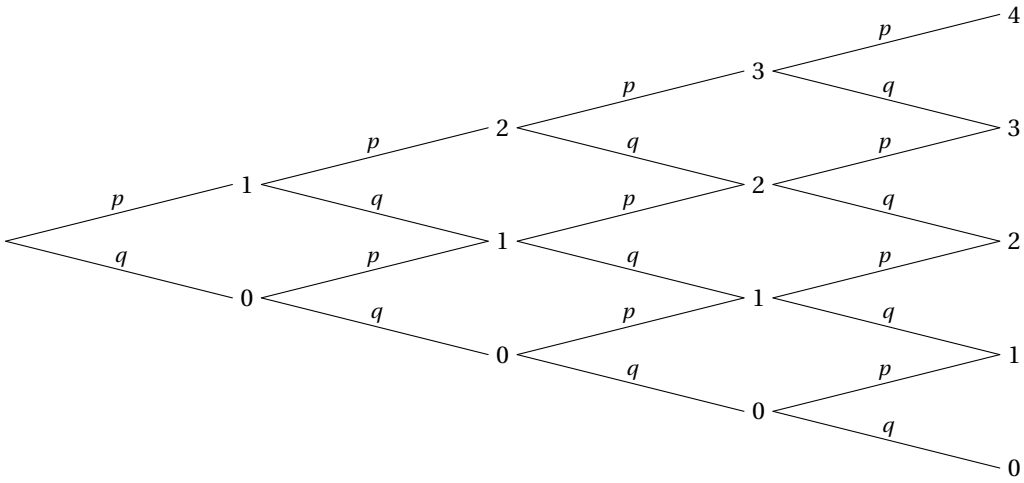


binomialtree(n)(l,h) tree describing the binomial distribution with n trials. `l` is the length of the first edges and `h` is the space between two final nodes. It uses `bernoullisuccesprob` and `bernoullifailureprob` but `bernoulliscalebranch` is set to `1`.

binomialtreeL(n)(L,H) same as above where `L` is the whole width of the tree and `H` its height.

Example 28

```
beginfig(28)
draw binomialtree(4)(3cm,1.5cm);
endfig;
```



7 Embedded code in LaTeX files

You can embed your code in LaTeX files.

7.1 With pdftex

Using emp package

```
pdflatex myfile.tex
mpost myfile.mp
mpost myfile.mp
pdflatex myfile.tex
```

```
\documentclass{article}
\usepackage{emp}
\usepackage{ifpdf}
 \ifpdf % allows pdflatex compilation
  \DeclareGraphicsRule{*}{mps}{*}{}
 \fi
\begin{document}
\begin{empfile}
\begin{empcmds}
 input mptrees;
\end{empcmds}
\begin{emp}(0,0)
 draw tree[1][1](3cm,3cm)(...);
\end{emp}
\end{empfile}
\end{document}
```

Using mpgraphics package

```
pdflatex -shell-escape myfile.tex
```

```
\documentclass{article}
\usepackage[runs=2]{mpgraphics}
\begin{document}
\begin{mpdefs}
 input mptrees;
\end{mpdefs}
\begin{mpdisplay}
 draw tree[1][1](3cm,3cm)(...);
\end{mpdisplay}
\end{document}
```

7.2 With luatex

Using LuaLaTeX

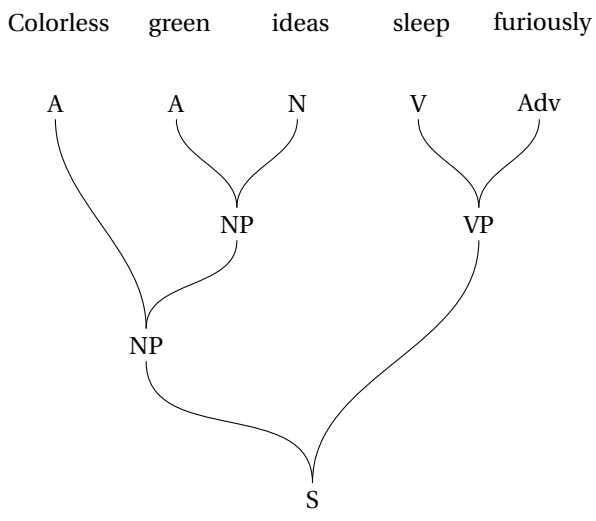
```
lualatex myfile.tex
```

```
\documentclass{article}
\usepackage{fontspec}
\usepackage{luamplib}
\begin{document}
\everymplib{input mptrees;}
\begin{mplibcode}
beginfig(1);
 draw tree[1][1](3cm,3cm)("$A$","$p$","$B$","$q$");
endfig;
\end{mplibcode}
\end{document}
```

8 Examples

Example 29

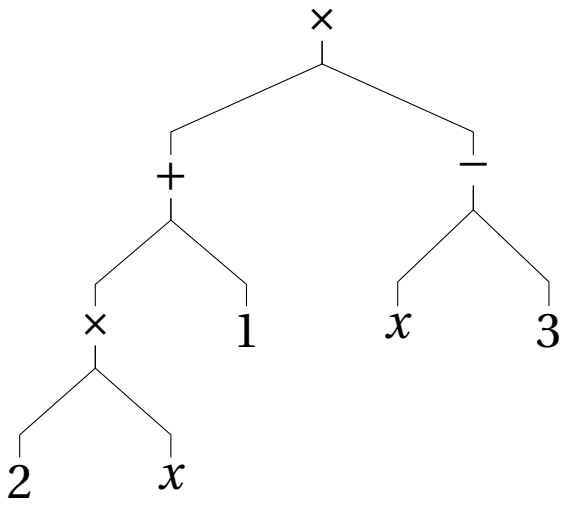
```
beginfig(29)
u:=0.4cm;
branchtype:="curve";
dirlabel:=90;
abscoord:=true;
endlabelspace:=0.5cm;
draw startlabel("S");
draw tree[1][1]((-5.5u,4u),(5.5u,8u))("NP","","VP","");
draw tree[2][1]((-8.5u,12u),(-2.5u,8u))("A","","NP","");
draw tree[2][2]((3.5u,12u),(7.5u,12u))("V","","Adv","");
draw tree[3][2]((-4.5u,12u),(-0.5u,12u))("A","","N","");
draw endlabel[3][1]("Colorless");
draw endlabel[4][1]("green");
draw endlabel[4][2]("ideas");
draw endlabel[3][3]("sleep");
draw endlabel[3][4]("furiously");
endfig;
```



Example 30

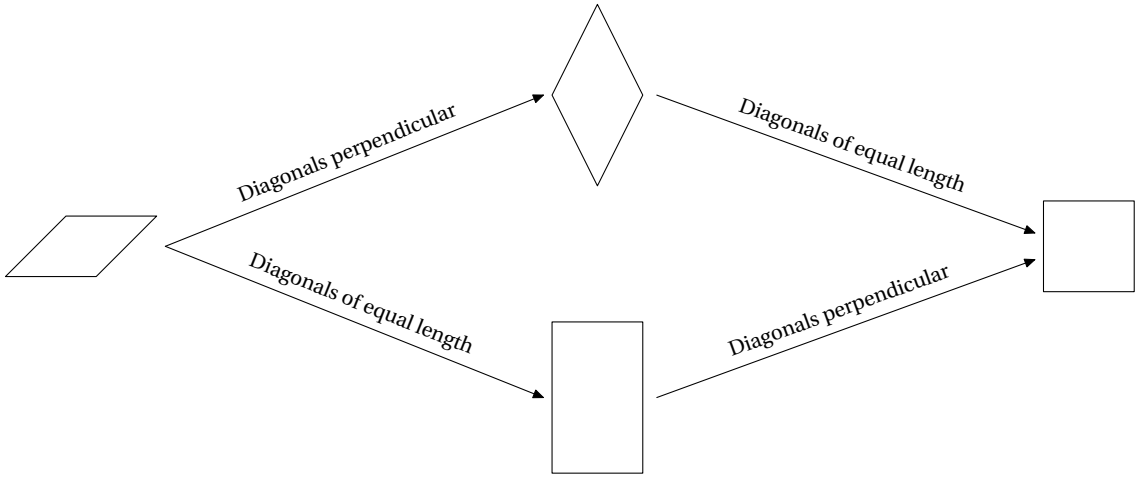
```
beginfig(30)
u:=1cm;
branchtype:="broken";
dirlabel:=-90;
abscoord:=true;
scaleev:=2;
label.top(textext("\Large Tree diagram of $(2x+1)(x-3)$"),(0,1cm));
draw startlabel("$\times$");
draw tree[1][1]((-2u,-1.5u),(2u,-1.5u))("$+$","","$-$","");
draw tree[2][1]((-3u,-3.5u),(-1u,-3.5u))("$\times$","","$1$","");
draw tree[2][2]((1u,-3.5u),(3u,-3.5u))("$x$","","$3$","");
draw tree[3][1]((-4u,-5.5u),(-2u,-5.5u))("$2$","","$x$","");
endfig;
```

Tree diagram of $(2x+1)(x-3)$



Example 31

```
beginfig(31)
posprob:=0.5;
typeprob:=3;
shiftev:=1.5cm;
edgearrow:=true;
u:=0.2cm;
vardef paral = ((2,-2)--(6,2)--(0,2)--(-4,-2)--cycle) scaled u enddef;
vardef rhombus = ((3,0)--(0,6)--(-3,0)--(0,-6)--cycle) scaled u enddef;
vardef rectangle = ((3,5)--(-3,5)--(-3,-5)--(3,-5)--cycle) scaled u enddef;
vardef square = ((3,3)--(-3,3)--(-3,-3)--(3,-3)--cycle) scaled u enddef;
draw startlabel(paral);
draw tree[1][1](5cm,4cm)(rhombus,"Diagonals perpendicular",%
                         rectangle,"Diagonals of equal length");
endedgeshift:=5;
draw tree[2][1]((5cm,-2cm))("","Diagonals of equal length");
draw tree[2][2]((5cm,2cm))(square,"Diagonals perpendicular");
endfig;
```


Diagonals perpendicular
Diagonals of equal length
Diagonals of equal length
Diagonals perpendicular