Part I

Trees

1 Overview

The first aim of this package is to simplify the drawing of probability trees (or other kind of "tree") with \textsc{MetaPost}. It provides one main command and several parameters to control the output.

It can be used in standalone files with two compilations (\texttt{latexmp} package is loaded) but also with \texttt{Lualatex} and \texttt{luamplib} package.
Probability tree located in column \( i \) and row \( j \) (see figure below). \( \text{dim}_1, \text{dim}_2, \ldots \) can be numerics or pairs and control the dimension of the tree. \( \text{ev}_1, \text{prob}_1, \ldots \) can be strings or pictures and will be printed (using \texttt{latexmp} if strings) at the end of the edge (the event) and above the edge (the probability).

\[
\begin{array}{c}
\text{tree}[i][j]\{\text{<dim}_1, \text{<dim}_2, \ldots}\}\{\text{<ev}_1, \text{<prob}_1, \text{<ev}_2, \text{<prob}_2, \ldots}\}
\end{array}
\]

Note that you can use these commands inside any \texttt{beginfig();...endfig;} but sometimes, for some constructions, they need to be enclosed between \texttt{begintree} and \texttt{endtree} commands. Such commands are indicated with a margin note.

## 2 Trees

### 2.1 Different kinds of trees

Regular tree where \texttt{width} is the horizontal width of the tree and \texttt{vspace} the vertical space between two consecutive nodes.

**Exemple 1**

\[
\begin{array}{c}
\text{beginfig(1);} \\
\text{draw tree}[1][1](4\text{cm},2.5\text{cm})\{"A_1","\text{\nicefrac{1}{3}}","A_2","\text{\nicefrac{2}{3}}"\}; \\
\text{draw tree}[2][1](3\text{cm},1.5\text{cm})\{"B","\text{\nicefrac{1}{4}}","C","\text{\nicefrac{3}{4}}"\}; \\
\text{draw tree}[2][2](3\text{cm},1\text{cm})\{"D","p","E","q","F","r"\}; \\
\text{endfig;}
\end{array}
\]
tree[<i>][<j>](<width>,<vsp1>,<vsp2>,...)(<ev1>,<p1>,<ev2>,<p2>,...)

Tree where width is the horizontal width of the tree while each vsp indicates the vertical space between the node and the origin of the tree.

Exemple 2

\begin{verbatim}
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;
\end{verbatim}

Exemple 3

\begin{verbatim}
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;
\end{verbatim}

Tree where pair1, pair2... indicate the coordinates of each node from the origin of the tree.
2.2 Simple trees

Same as previous except that there are no probabilities.

Exemple 4

```latex
beginfig(4);
  draw stree[1][1](100,50)("$A$","$B$","$C$"IAS);  
  draw stree[2][1](80,25)("$A$","$B$"IAS);
  draw stree[2][2](80,25)("$A$","$B$"IAS);
  draw stree[2][3](80,25)("$A$","$B$"IAS);
endfig;
```

2.3 Start and end labels

**startlabel<(<s>)>** picture

Prints \(s\) (can be a string or a picture) at the origin of the tree.

Exemple 5

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

**endlabel<(<i>)<(<j>)>(<s>)>** picture

Prints \(s\) at the end of a branch. The space between the previous label and \(s\) is controlled by the numeric endlabelspace which defaults to 1cm.

Exemple 6

```latex
beginfig(6);
  draw startlabel("$S$"IAS);
  draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$"IAS);
  draw tree[2][2](2cm,1cm)("$A$","$p$","$B$","$q$"IAS);
  draw endlabel[2][1]("$SA$"IAS);
  draw endlabel[3][1]("$SBA$"IAS);
  draw endlabel[3][2]("$SBB$"IAS);
endfig;
```
3 Direction

dirtree numeric, default: 0

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

Exemple 7

beginfig(7);
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;

Exemple 8

beginfig(8);
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;
All the trees are viewed as “horizontal” trees, so the space between two subtrees is horizontal too. With dirtree, the whole (horizontal) tree is rotated. But if the tree is designed vertically, spacing is wrong. In this case, one can use dirlabel to indicate the orientation of the tree.

**Exemple 9**

```latex
beginfig(9);
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;
```

**Exemple 10**

```latex
beginfig(10);
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

The origin of each tree is located at the right side of the bounding box of the previous event name. Thus different subtrees may begin at different places. The numeric \texttt{shift\textsubscript{ev}}, if positive, indicates the fixed horizontal space between the end of the edges and the beginning of following subtrees. It can be used inside the first set of parameters of the tree (see example below) or as a global variable.

\begin{example}
\begin{verbatim}
beginfig(11);
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,\texttt{shift\textsubscript{ev}=1.5cm})("Yes","$p$","No","$q$","Maybe","$r$");
draw tree[3][1](50,20)("$B$","$b$","$C$","$c$");
draw tree[3][2](50,20)("Yes","$p$","No","$q$","Maybe","$r$");
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;
\end{verbatim}
\end{example}

\texttt{abscoord boolean, default: false}

With the boolean \texttt{abscoord} set to \texttt{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.
5 Parameters

All following parameters can be changed globally before drawing the tree or changed locally inside the first set of parameters:

\[
\text{scaleev:=2;}
\text{draw tree[1][1](3cm,2cm)(...);}
\text{draw tree[2][1](3cm,2cm)(...);}
\]

or

\[
\text{draw tree[1][1](3cm,2cm,"scaleev:=2")(...);}
\text{draw tree[2][1](3cm,2cm)(...);}
\]

In the first case, \text{scaleev} is changed globally while in the second case, the change only applies to the first tree.

5.1 Event

\text{scaleev numeric, default: 1}

Numeric controlling the scale of the label at the end of the edge (the event).

Exemple 13

\text{beginfig(13);}
\text{scaleev:=2;}
\text{draw stree[1][1](3cm,2cm)("A","B");}
\text{endfig;}

\[A\quad B\]

\[\begin{align*}
\text{Exemple 12} \\
\text{beginfig(12);}
\text{abscoord:=true;}
\text{draw tree[1][1](3cm,2cm)("$A$","$p$","Blabla","$q$");}
\text{draw tree[2][1](7cm,1.5cm),(7cm,0.5cm)("$A$","$p$","$B$","$q$");}
\text{draw tree[2][2](7cm,-0.5cm),(7cm,-1.5cm)("$A$","$p$","$B$","$q$");}
\text{endfig;}
\end{align*}\]
### nodeformat

String that indicates how the events are printed (the shape of path around the event). Possible values are (for now) "bbox", "circle", "superellipse".

**Exemple 14**

```latex
defbeginfig(14);
nodeformat:="bbox";
draw stree[1][1](3cm,2cm)("$A$","$B$" );
endfig;
```

### nodelinecolor

Color of the path around the node

**Exemple 15**

```latex
beginfig(15);
nodelinecolor:=(0.8,0,0);
nodebgcolor:=(1,0.5,0.5);
nodefgcolor:=white;
draw stree[1][1](3cm,2cm)("$A$","$B$" );
endfig;
```

### nodebgcolor

Color of the background of the region delimited by the previous path

**Exemple 16**

```latex
beginfig(16);
nodefgcolor:=(0.7,0.4,0.7);
draw stree[1][1](3cm,2cm)("$A$","$B$" );
endfig;
```

### leaveformat

String that indicates how the events are printed (the shape of path around the event). Possible values are (for now) "bbox", "circle", "superellipse" and "none".

#### 5.2 Leaves

You may want to format the leaves in a different way from the nodes. A tree using the following parameters must be enclosed in a `begintree;...endtree; "environment"`. 
Exemple 17

beginfig(17);
begintree;
leaveformat:="bbox";
\draw stree[1][1](100,45)("$A$","$B$" );
\draw stree[2][1](80,30)("$C$","$D$" );
\draw stree[3][2](65,20)("$E$","$F$" );
endtree;
endfig;

leavelinecolor color, default: black

Color of the path around the leave

leavebgcolor color, default: white

Color of the background of the region delimited by the previous path

leavefgcolor color, default: black

Color of the text.

Exemple 18

beginfig(18);
begintree;
nodelinecolor:=(0.8,0,0); nodebgcolor:=(1,0.5,0.5); nodefgcolor:=white;
leaveformat:="bbox";
leavebgcolor:=(0.3,1,1); leavefgcolor:=red;
\draw stree[1][1](100,45)("$A$","$B$" );
\draw stree[2][1](80,30)("$C$","$D$" );
\draw stree[3][2](65,20)("$E$","$F$" );
endtree;
endfig;

Note that nodeformat applies to both nodes and leaves. To avoid formatting the leaves, use the value "none" for leaveformat.
5.3 Probability

**probformat**

String that indicates how the probabilities are printed (the shape of path around the probability). Possible values are (for now) "bbox", "circle", "superellipse".

**problinicolor**

Color of the path around the probability

**probbgcolor**

Color of the background of the region delimited by the previous path.

**probfgcolor**

Color of the text.
Exemple 21

```
beginfig(21);
probformat:="circle";
problinecolor:=(0.8,0,0);
probbgcolor:=(1,0.5,0.5);
probfgcolor:=white;
draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$"); 
endfig;
```

Exemple 22

```
beginfig(22);
probfgcolor:=blue;
draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$"); 
endfig;
```

**scaleprob**

Numeric controlling the scale of the label above the edge (the probability).

Exemple 23

```
beginfig(23);
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.

Exemple 24

```
beginfig(24);
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 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).

Exemple 25

```
beginfig(25);
typeprob:=2;
draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","1-p$"); 
endfig;
```
Exemple 26

```latex
beginfig(26);
typeprob:=3;
draw tree[1][1](3cm,2cm)("A", "p", "B", "1-p");
endfig;
```

Exemple 27

```latex
beginfig(27);
typeprob:=4;
draw tree[1][1](3cm,2cm)("A", "p", "B", "1-p");
endfig;
```

**proboffset**

Numeric, default: 3bp

Numeric controlling the amount by which the label above the edge is offset.

Exemple 28

```latex
beginfig(28);
draw tree[1][1](3cm,3cm)("A", "p+q+r", "B", "s");
endfig;
```

Exemple 29

```latex
beginfig(29);
proboffset:=6bp;
draw tree[1][1](3cm,3cm)("A", "p+q+r", "B", "s");
endfig;
```

5.4 Edge

**linewidth**

numeric, default: 0.5bp

Width of the lines.

**linecolor**

color, default: black

Color of the lines.

Exemple 30

```latex
beginfig(30);
linewidth:=1.5;
linecolor:=blue;
draw tree[1][1](3cm,2cm)("A", "p", "B", "q");
endfig;
```
**endedgeshift**

Vertical space added at the end of the edge. Useful when various edges end at the same point.

**Exemple 31**

```plaintext
beginfig(31);
  draw startlabel("$S$");
  draw tree[1][1]((3cm,-1cm))("$A$","$p$");
endfig;
```

**Exemple 32**

```plaintext
beginfig(32);
  endedgeshift:=10;
  draw startlabel("$S$");
  draw tree[1][1]((3cm,-1cm))("$A$","$p$");
endfig;
```

**edgearrow**

Boolean, default: false

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

**Exemple 33**

```plaintext
beginfig(33);
  edgearrow:=true;
  draw tree[1][1](3cm,2cm)("$A$","$p$","$B$","$q$");
endfig;
```

**branchtype**

String, default: "segment"

String which indicates the shape of the edge. Possible values are segment, curve, broken. Note that double quotes have to be replaced by single quotes when this parameter is changed locally inside the `tree` macro.

**tenscurve**

Numeric, default: 0

If string `branchtype` is set to curve, `tenscurve` indicates the “tension”. When sets to 1, the curve is a segment.

**Exemple 34**

```plaintext
beginfig(34);
  branchtype:="curve";
  draw stree[1][1](3cm,3cm)("$A$","$B$");
endfig;
```
Exemple 35

```
beginfig(35);
  draw stree[1][1](3cm,3cm,"branchtype='curve'")
    ("$A$","$B$");
endfig;
```

Exemple 36

```
beginfig(36);
  branchtype="curve";
  tenscurve:=0.5;
  draw stree[1][1](3cm,3cm)("$A$","$B$");
endfig;
```

**brokenlineratio** numeric, default: 0.2

If string `branchtype` is set to `broken`, `brokenlineratio` indicates the ratio between the length of the first segment of the broken line and the total length of the horizontal space.

Exemple 37

```
beginfig(37);
  branchtype="broken"
  draw stree[1][1](3cm,3cm)("$A$","$B$");
endfig;
```

Exemple 38

```
beginfig(38);
  branchtype="broken"
  posprob:=0.8;
  brokenlineratio:=0.5;
  draw stree[1][1](3cm,3cm)("$A$","$B$");
endfig;
```

Exemple 39

```
beginfig(39);
  branchtype="broken"
  posprob:=0.8;
  brokenlineratio:=0.75;
  draw stree[1][1](3cm,3cm)("$A$","$B$");
endfig;
```
6 Regular trees

6.1 Ordinary regular trees

\begin{verbatim}
regulartree(<n>)(<l>,<h>)(<ev1>,<prob1>,<ev2>,<prob2>,...) picture
\end{verbatim}

Tree describing the repetition of \( n \) identical and independent random experiments. \( l \) is the horizontal length of the first edges and \( h \) is the vertical space between two leaves.

\begin{verbatim}
scalebranch numeric, default : 0.8
\end{verbatim}

Ratio between edges width of consecutive level.

**Exemple 40**

```
beginfig(40);
draw regulartree(2)(3cm,0.7cm)
   ("$A$","$p$","$B$","$q$","$C$","$r$"); 
endfig;
```

```
  A
 / \
 p q r
 B C
```

Note that you can change variable values inside the first set of parameters.

**Exemple 41**

```
beginfig(41);
draw regulartree(2)(3cm,0.7cm,"typeprob:=2")
    ("$A$","$p$","$B$","$q$","$C$","$r$"); 
endfig;
```

```
  A
 / \
 p q r
 B C
```

6.2 Binomial trees

\begin{verbatim}
bernoulliprocess(<n>)(<l>,<h>)(<ev1>,<prob1>,<ev2>,<prob2>) picture
\end{verbatim}

Tree describing the Bernoulli process with \( n \) trials. \( l \) is the horizontal length of the first edges and \( h \) is the vertical space between two final nodes. If the last set of parameters is omitted, the values are set according to the following parameters.

\begin{verbatim}
bernoulliprocessL(<n>)(<L>,<H>)(<ev1>,<prob1>,<ev2>,<prob2>) picture
\end{verbatim}

Same as above where \( L \) is the whole width of the tree and \( H \) its height.
Several parameters control the output:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bernoullisuccessevent</code></td>
<td>&quot;$S$&quot;</td>
<td>String printed at every node representing a success.</td>
</tr>
<tr>
<td><code>bernoullifailureevent</code></td>
<td>&quot;$\overline{S}$&quot;</td>
<td>String printed at every node representing a failure.</td>
</tr>
<tr>
<td><code>bernoullisuccessprob</code></td>
<td>&quot;$p$&quot;</td>
<td>String printed above every edge representing a success.</td>
</tr>
<tr>
<td><code>bernoullifailureprob</code></td>
<td>&quot;$q$&quot;</td>
<td>String printed above every edge representing a failure.</td>
</tr>
<tr>
<td><code>bernoulliscalebranch</code></td>
<td>0.8</td>
<td>Ratio between width of consecutive edges.</td>
</tr>
</tbody>
</table>

**Exemple 42**

```plaintext```
beginfig(42);
draw bernoulliprocess(3)(3cm,0.7cm)();
endfig;
```

**Exemple 43**

```plaintext```
beginfig(43);
draw bernoulliprocess(3)(3cm,0.7cm,"typeprob:=2;")
("$A$","$0.7$","$B$","$0.3$");
endfig;
```
Exemple 44

\[
\begin{align*}
\text{beginfig}(44); \\
\text{typeprob}:=4; \\
\text{bernoullisuccessevent}:="A"; \\
\text{bernoullifailureevent}:="B"; \\
\text{bernoullisuccessprob}:="0.7"; \\
\text{bernoullifailureprob}:="0.3"; \\
\text{draw bernoulliprocess}(3)(3\,\text{cm},0.7\,\text{cm})(); \\
\text{endfig};
\end{align*}
\]

\[
\begin{tikzpicture}
    \node (A) at (0,0) {$A$};
    \node (B) at (0,-1) {$B$};
    \node (C) at (1,0) {$A$};
    \node (D) at (1,-1) {$B$};
    \draw (A) -- (C) node[midway] {0.7};
    \draw (A) -- (D) node[midway] {0.3};
    \draw (B) -- (C) node[midway] {0.7};
    \draw (B) -- (D) node[midway] {0.3};
\end{tikzpicture}
\]

\text{binomialtree}(<n>)(<l>,<h>) \text{picture}

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 \text{bernoullisuccesprob} and \text{bernoullifailureprob} but \text{bernoulliscalebranch} is set to 1.

\text{binomialtreeL}(<n>)(<L>,<H>) \text{picture}

Same as above where \( L \) is the whole width of the tree and \( H \) its height.

Exemple 45

\[
\begin{align*}
\text{beginfig}(45); \\
\text{draw binomialtree}(4)(3\,\text{cm},1.5\,\text{cm}); \\
\text{endfig};
\end{align*}
\]

\[
\begin{tikzpicture}
    \node at (0,0) {$0$};
    \node at (1,0) {$1$};
    \node at (2,0) {$2$};
    \node at (3,0) {$3$};
    \node at (4,0) {$4$};
    \node at (0,1) {$q$};
    \node at (0,-1) {$p$};
    \node at (1,1) {$q$};
    \node at (1,-1) {$p$};
    \node at (2,1) {$q$};
    \node at (2,-1) {$p$};
    \node at (3,1) {$q$};
    \node at (3,-1) {$p$};
    \node at (4,1) {$q$};
    \node at (4,-1) {$p$};
\end{tikzpicture}
\]

\text{tree}[,\ldots] \text{picture}

The following commands are experimental and need to be enclosed in a \text{begintree;...endtree; “environment”}.

7 “Calculated” trees

When the first set of parameters is left empty, the dimensions of the tree are calculated. The calculations use the parameters described below.
Same as above for “simple” trees.

\textbf{Exemple 46}

```plaintext
beginfig(46);
begintree;
\texttt{draw startlabel("$S$"});
\texttt{draw stree[1][1]()("$A$","$B$","$C$"});
\texttt{draw stree[2][1]()("$D$","$E$","$F$");}
\texttt{draw stree[2][3]()("$D$","$E$"};
\texttt{draw stree[3][2]()("$G$","$H$"};
\texttt{draw stree[3][4]()("$G$","$H$"};
\texttt{endtree;}
endfig;
```

\begin{verbatim}
widthbranch numeric, default: 3.5cm
\end{verbatim}

Horizontal width of the first level tree.

\begin{verbatim}
gapnode numeric, default: 0.7cm
\end{verbatim}

Minimal vertical space between two nodes of the last level trees.

\begin{verbatim}
scalebranch numeric, default: 0.8
\end{verbatim}

Ratio between edges width of consecutive level.
Exemple 47

\begin{verbatim}
beginfig(47);
beginintree;
widthbranch:=4cm;
scalebranch:=1;
gapnode:=1cm;

draw startlabel("$S$");
draw stree[1][1](()("$A$","$B$","$C$"));
draw stree[2][1](()("$D$","$E$","$F$"));
draw stree[2][3](()("$D$","$E$"));
draw stree[3][2](()("$G$","$H$"));
draw stree[3][4](()("$G$","$H$"));
endtree;
endfig;
\end{verbatim}

8 Examples

Exemple 48

\begin{verbatim}
beginfig(48);
u:=0.4cm;
branchtype:="curve";
dirlabel:=90;
abscoord:=true;
endlabelspace:=0.5cm;
draw startlabel("$S$");
draw stree[1][1]((-5.5u,4u),(5.5u,8u))("NP","VP");
draw stree[2][1]((-8.5u,12u),(-2.5u,8u))("A","NP");
draw stree[2][2]((3.5u,12u),(7.5u,12u))("V","Adv");
draw stree[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;
\end{verbatim}
Colorless green ideas sleep furiously

Exemple 49

beginfig(49);
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 stree[1][1]((-2u,-1.5u),(2u,-1.5u))("+$","-$");
draw stree[2][1]((-3u,-3.5u),(-1u,-3.5u))("$\times$","$1$" quả

draw stree[2][2]((1u,-3.5u),(3u,-3.5u))("$x$","$3$" quả

draw stree[3][1]((-4u,-5.5u),(-2u,-5.5u))("$2$","$x$" quả
endfig;

Tree diagram of $(2x + 1)(x - 3)$
Exemple 50

beginfig(50);
  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;
Exemple 51

\begin{fig}
\dirtree:=-90;
\branchtype:="curve"; \tenscurve:=0.75;
\linewidth:=1; \linecolor:=(0.2,0.2,0.7);
\widthbranch:=1cm; \scalebranch:=0.9;
gapnode:=1cm;
\leaveformat:="bbox";
\nodeformat:="superellipse"; \nodebgcolor:=(0.6,0.6,1);
\beginintree
\text{label.top}(\text{texext(\LaTeX\_Huffman\_tree\_(source\_Wikipedia)})\(,\)(0,1cm))
\draw \text{startlabel}(36);
\draw \text{stree}[1][1](()("20","16")\);
\draw \text{stree}[2][1](()("12","8")\);
\draw \text{stree}[2][2](()("8","8")\);
\draw \text{stree}[3][1](()("'|7","5")\);
\draw \text{stree}[3][2](()("4","4")\);
\draw \text{stree}[3][3](()("4","a'|4")\);
\draw \text{stree}[3][4](()("4","e'|4")\);
\draw \text{stree}[4][2](()("f'|3","2")\);
\draw \text{stree}[4][3](()("s'|2","h'|2")\);
\draw \text{stree}[4][4](()("2","i'|2")\);
\draw \text{stree}[4][5](()("m'|2","t'|2")\);
\draw \text{stree}[4][7](()("2","n'|2")\);
\draw \text{stree}[5][2](()("1'|1","x'|1")\);
\draw \text{stree}[5][5](()("p'|1","x'|1")\);
\draw \text{stree}[5][9](()("u'|1","o'|1")\);
\endintree
\end{fig}

Huffman tree (source Wikipedia)
9 Overview

This package can also be used to draw graphs. In this case, you need to define the nodes by their coordinates and indicate the edges to be drawn. Various commands allow you to draw graphs quickly, but it is also possible to finely control the drawing of each node and edge.

10 Nodes

10.1 Definition

\texttt{defnodes(<pair1>,<pair2>,...)}

The easiest way to define the nodes: just put a list of pairs. This command does not draw anything, it just defines an array of pairs corresponding to the coordinates. By default the array name is $A[]$.

\texttt{defnodename \ string, default: "A"}

Default node array name.

\texttt{nodename \ string, default: "array"}

String indicating how the nodes are stored and labels are drawn:
- with {\texttt{nodename}="array"}, nodes are stored in $A[]$ and labelled $A1, A2,...$
- with {\texttt{nodename}="Alph"}, nodes are stored and labelled $A, B,...$
- with {\texttt{nodename}="alph"}, nodes are stored and labelled $a, b,...$
- with {\texttt{nodename}="arabic"}, nodes are stored in $A[]$, but are labelled $1, 2,...$

\texttt{defnode(<token>,<pair>)}

Nodes can be defined one by one. <token> is the node's name and <pair> its coordinates.

10.2 Drawing

\texttt{drawnodes.<pos>(<nodei>,<nodej>,...)}

Draw all the indicated nodes. These nodes can be designated either by name or by rank. If the list is empty, all the nodes are drawn. \texttt{pos} can be empty or \texttt{lft}, \texttt{rt}, etc. or an angle.

\begin{verbatim}
Exemple 52
beginfig(52);
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes(1,3);
endfig;
\end{verbatim}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{example52}
\caption{Example of graph with nodes and edges drawn.}
\end{figure}
Exemple 53
beginfig(53);
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes(A1,A3);
endfig;

Exemple 54
beginfig(54);
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;

Exemple 55
beginfig(55);
defnodename:="N";
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;

Exemple 56
beginfig(56);
nodename:="Alph";
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;

Exemple 57
beginfig(57);
nodename:="alph";
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;

Exemple 58
beginfig(58);
nodename:="arabic";
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;

\texttt{printnodename, default: true}

When set to false, node names are not printed.
Exemple 59

```plaintext
beginfig(59);
printnodename:=false;
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;
```

**nodeformat**

String that indicates how the nodes are printed (the shape of path around the node). Possible values are (for now) "bbox", "circle", "superellipse" and "square".

**nodewidth**

Node width...

Exemple 60

```plaintext
beginfig(60);
nodewidth:=1cm;
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;
```

```
\[ A_1 \quad A_2 \quad A_3 \]
```

Exemple 61

```plaintext
beginfig(61);
nodewidth:=0.2cm;
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes\bot();
endfig;
```

```
\[ A_1 \quad A_2 \quad A_3 \]
```

**naturalwidth**

When set to true, the width of the node adapts to the content.

Exemple 62

```plaintext
beginfig(62);
defnodename:="AA";
naturalwidth:=true;
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;
```

```
\[ AA_1 \quad AA_2 \quad AA_3 \]
```

**nodelinewidth**

Line width of the path around the node.
Color of the path around the node.

Color of the background of the region delimited by the previous path.

Color of the text.

**Exemple 63**

```plaintext
beginfig(63);
nodelinecolor:=(0.8,0,0);
nodebgcolor:=(1,0.5,0.5);
nodefgcolor:=white;
nodeformat:="superellipse";
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes();
endfig;
```

While `withcolor` applies to lines and labels, `withbgcolor` applies to the filling of nodes.

**Exemple 64**

```plaintext
beginfig(64);
nodelinecolor:=(0.8,0,0);
nodebgcolor:=(1,0.5,0.5);
nodefgcolor:=white;
nodeformat:="superellipse";
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnodes() withcolor black withbgcolor (0.6,0.6,1);
endfig;
```

The nodes can be drawn one by one, specifying the text to be displayed if necessary.

**Exemple 65**

```plaintext
beginfig(65);
defnodes((0cm,0cm),(2cm,0),(4cm,0));
drawnode[60](1);
drawnode(2, "$\sqrt{2}\$" ) withbgcolor 0.7 white;
drawnode. bot(A3,"Node");
endfig;
```
11 Edges

11.1 Undirected edges

\texttt{drawedges.<pos>(<(Na,Nb)>,<string1>,<(Nc,Nd)>,<string2>,...)(<angle1>,<angle2>)}

Draw undirected edges between Node Na and node Nb, between node Nc and node Nd,... Na, Nb, Nc, Nd... must be integers. You can't refer to the nodes with their name.
If present, the string is printed at the position indicated in relation to the middle of the edge (by default).
By default, the edge is a segment. If an angle is specified, it is added to the angle at the start of the edge and subtracted from the angle at the end of the edge. If two angles are specified, they are added to the start and end angles respectively.

\textbf{Exemple 66}

\begin{verbatim}
beginfig(66);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),(3,1))();
endfig;
\end{verbatim}

\textbf{Exemple 67}

\begin{verbatim}
beginfig(67);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),(3,1),"$p$"());
endfig;
\end{verbatim}

\textbf{Exemple 68}

\begin{verbatim}
beginfig(68);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges\texttt{.bot}((1,2),(3,1),"$p$"())();
endfig;
\end{verbatim}

\textbf{Exemple 69}

\begin{verbatim}
beginfig(69);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),(3,1))(30);
endfig;
\end{verbatim}

\textbf{Exemple 70}

\begin{verbatim}
beginfig(70);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),(3,1))(40,10);
endfig;
\end{verbatim}
The edges can be drawn one by one. In this case, Na, Nb can be the node name. The label or weight must be indicated in the second set of parameters.

**Exemple 71**

```plaintext
beginfig(71);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedge(1,2)();
drawedge.bot(1,3)("p");
drawedge(2,3)(30,"2");
endfig;
```

**11.2 Directed edges**

drawdiredges.<pos>((Na,Nb),(string1),(Nc,Nd),(string2),...)(<angle1>,<angle2>)

Draw directed edges from Node Na to node Nb, from node Nc to node Nd,... Na, Nb, Nc, Nd... must be integers. You can't refer to the nodes with their name.

If present, the string is printed at the position indicated in relation to the middle of the edge (by default).

By default, the edge is a segment. If an angle is specified, it is added to the angle at the start of the edge and subtracted from the angle at the end of the edge. If two angles are specified, they are added to the start and end angles respectively.

**Exemple 72**

```plaintext
beginfig(72);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawdiredges((1,2),"35",(3,1),"p")();
endfig;
```

**Exemple 73**

```plaintext
beginfig(73);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawdiredges.bot((1,2),(3,1),"p")(30);
endfig;
```
Exemple 74

```latex
beginfig(74);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawdiredge(1,2);)
drawdiredge.bot(1,3)("p");
drawdiredge(2,3)(30,"2");
endfig;
```

11.3 Loops

```latex
draw(dir).edge.<pos>(<Na>,<Na>)(<angle1>,<angle2>,<string>)
```

Previous commands can be used to draw loops. If present, angle1 indicates the angle of the loop with horizontal direction. Default value is 90.

Exemple 75

```latex
beginfig(75);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawdiredge(1,1);)
drawedge(3,3)(-150); 
drawdiredge.bot(2,2)(-90,"a");
endfig;
```

11.4 Parameters

**edgelinewidth**

numeric, default: 1

Width of the edge.

Exemple 76

```latex
beginfig(76);
edgelinewidth:=2;
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),(3,1),"p");
endfig;
```

**nodeedgeoffset**

numeric, default: 0

Space between the node and the edge.

Exemple 77

```latex
beginfig(77);
nodeedgeoffset:=0.3cm;
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),(3,1),"p");
endfig;
```
probformat  

String that indicates how the weights are printed (the shape of the path). Possible values are (for now) "bbox", "circle", "superellipse" and "square".

problinewidth  

Color of the path around the weight.

probbgcolor  

Color of the background of the region delimited by the previous path.

probfgcolor  

Color of the text.

Exemple 78

```plaintext
beginfig(78);
problinewidth:=(0.8,0,0);
probbgcolor:=(1,0.5,0.5);
probfgcolor:=white;
probformat:="square";
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),"35",(3,1),"$p$"());
endfig;
```

```
\begin{tikzpicture}
  \node (A1) at (0,0) {$A_1$};
  \node (A2) at (2,2) {$A_2$};
  \node (A3) at (4,0) {$A_3$};
  \draw (A1) -- (A2) node[midway] {$35$};
  \draw (A1) -- (A3) node[midway] {$p$};
  \node[red] at (1.5,1.5) {35};
\end{tikzpicture}
```

Position of the weight on the edge.

Exemple 79

```plaintext
beginfig(79);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
posprob:=0.75;
drawnodes();
drawedges((1,2),"35",(3,1),"$p$"());
endfig;
```

```
\begin{tikzpicture}
  \node (A1) at (0,0) {$A_1$};
  \node (A2) at (2,2) {$A_2$};
  \node (A3) at (4,0) {$A_3$};
  \draw (A1) -- (A2) node[midway] {$35$};
  \draw (A1) -- (A3) node[midway] {$p$};
  \node[red] at (1.5,1.5) {35};
\end{tikzpicture}
```

Position of arrowhead on the edge.
Example 80

beginfig(80);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
edgearrowpos:=0.5;
drawnodes();
drawdiredges.bot((1,2),(3,1),"$p$")();
endfig;

\textbf{loopangle}

Loop direction. It can be specified in the command parameters.

Example 81

beginfig(81);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
loopangle:=-90;
drawnodes();
drawdiredges((1,1),(2,2),(3,3))();
endfig;  

\textbf{loopstartangle}

Angle between the path direction at node center and loop direction.

Example 82

beginfig(82);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
loopstartangle:=90;
drawnodes();
drawdiredges((1,1),(2,2),(3,3))(-90);
endfig;  

\textbf{loopsize}

Loop size.
12 Complete graphs

\texttt{\texttt{draw(dir)graph.<pos>\texttt{((Na,Nb)},\texttt{<(string1>,<(string2),\ldots\texttt{)}(\texttt{<angle1>,<angle2>)}}\}

These commands are shortcuts for drawing both nodes and edges.

13 Grids

To facilitate node placement, use the following command to draw a grid.

\texttt{\texttt{drawgrid(<xmin>,<ymin>,<xmax>,<ymax>)(<unit>,<step>)}}

Command for drawing a grid. Minimum values are optional and equal to 0 by default. The unit and step values are also optional and default to 1 cm and 1 respectively.
Exemple 88

beginfig(88);
drawgrid(-1,-1,5,3)(1cm,0.5);
defnodes((0cm,0cm),(2cm,2cm),(4cm,0));
drawnodes();
drawedges((1,2),(3,1));
endfig;

Exemple 89

beginfig(89);
u:=0.75cm;
drawgrid(-1,-1,5,3)(u);
defnodes((0,0),(2u,2u),(4u,0));
drawnodes();
drawedges((1,2),(3,1));
endfig;