The PracTEX Journal, 2007, No. 1
Article revision 2007/02/20

Create Trees and Figures in Graph Theory with PSTricks

Manjusha S. Joshi
Bhaskaracharya Institute in Mathematics, Pune, India

Abstract Drawing trees and figures in the subject Graph Theory (Mathematics) is a requirement for researchers and teachers. This include loops, arcs, nodes, weights for edges. This article aims to get started with PSTricks by keeping two commands in mind viz. `pstree` and `psmatrix` with their most useful options with the help of which a reader can draw neat tree diagrams, loops, can label nodes, add weights to edges. After completing the diagram, one can add these diagrams to TEX file. So working knowledge about drawing figures in graph theory can be gained.

Contents

1 Preliminaries

Prerequisites: Basic knowledge of LaTeX.

Expectations: After going through this article newbee to PSTricks can also create his/her own figure files and will be able to insert it in his/her TeX document.

System requirement: LaTeX with PSTricks packages

Web site: You can get the required package and information from

`http://tug.org/PSTricks/`

Preamble should include packages: pstricks, pst-node, pst-tree

2 psmatrix

In the command `\psmatrix` the word 'matrix' is involved. Let us have a look at matrix. We write a matrix as $A_{3\times4}$, which indicates that the matrix A has 3 rows and 4 column. These are called as dimensions of the matrix.

$$\begin{matrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{matrix}$$

Each element in the matrix can be identified with, *which row?* and *which column?* e.g. a_{23} indicates that the element is in the 2nd row and 3rd column. So the order is row number followed by column number.

Now, let us have a look at the environment `\psmatrix`

```
$\psmatrix

\endpsmatrix
$
```

Though, this environment doesn't start with `\begin`, it ends with `\end` command. Note this difference with other TeX environments.

To use these special PSTricks commands one requires to include `\usepackage{pstricks,pst-node,pst-tree}`. Here is the minimum code that includes the necessary preamble.

```
\documentclass{article}
\usepackage{graphics,graphicx}
\usepackage{pstricks,pst-node,pst-tree}
\begin{document}
$
\psmatrix
u&v&z\\
x&y
\endpsmatrix
$
\end{document}
```

The out put will look exactly like this:

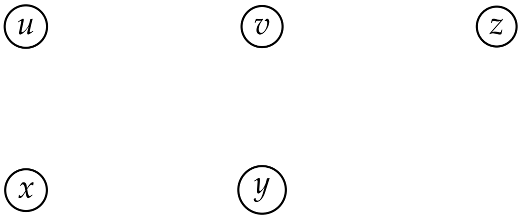
$$\begin{array}{ccc} u & v & z \\ & & \\ x & y & \end{array}$$

Note that the vertical and horizontal distances are by default. We can control them by adding optional argument to psmatrix command with `\psmatrix[colsep=2.5cm,rowsep=1.5cm]` thus we ask to keep column separation as 2.5cm and row separation as 1.5cm.

Also, frequently we like to denote these points with small circles around the letters like ⓐ. This can be obtained by `\psmatrix[mnode=circle]`, which means choose node of type circle. So after adding these two things our code will be:

```
\documentclass{article}
\usepackage{graphics,graphicx}
\usepackage{pstricks,pst-node,pst-tree}
\begin{document}
$
\psmatrix[colsep=2.5cm,rowsep=1.5cm,mnode=circle]
&u&v&z\\
&x&y
\endpsmatrix
$
\end{document}
```

The output of the code looks like:



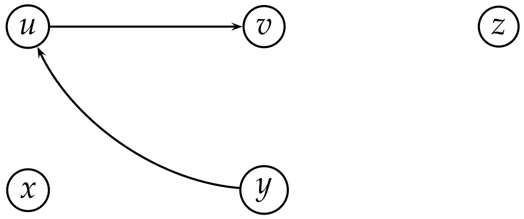
Such an output can be used to draw the vertices of a graph.

2.1 Types of edges: lines, arcs and loops

Now we would like to join vertices by edges which can be of types: Lines, arc or loop. For joining the vertices remember the matrix way to locate an element. The same way will be followed to refer to an element.

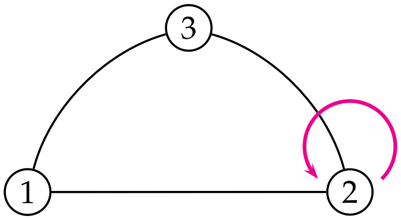
Now we want to join elements u and v in the above figure. What is the place of u? It is in the first row and the first column, so refer to it by (1,1), now since v is in the first row and the second column, it is referred to by (1,2); remember the order (row,column). Finally use the command `\ncline{1,1}{1,2}` to draw line segment between these two nodes. Many times just the line is not sufficient, we are require to denote the direction also. For this, add `{<->}`, from whom towards whom? If $v \to u$ is the direction then order should change to `{1,2}{1,1}`.

```
\documentclass{article}
\usepackage{graphics,graphicx}
\usepackage{pstricks,pst-node,pst-tree}
\begin{document}
$
\psmatrix[colsep=2cm,rowsep=1cm,mnode=circle]
u&v\\
z\\
x&y
\ncline{<->}{1,2}{1,3}
\ncarc[arcangle=-30]{<-}{1,1}{2,2}
\endpsmatrix
$
\end{document}
```



`\ncarc[arcangle=-30]{<->}{1,1}{2,2}` this command is useful to draw a circular edge between the element in first row and first column to the element in second row second column.

To obtained a loop for a node `\nccircle{->}{1,1}{.5cm}` this command is used. If you want to show the direction, you can add bracket `{->}` followed by node (referred by its row,column) followed by radius.



```
 $
\psmatrix[colsep=1.5cm,rowsep=1.5cm,mnode=circle]
&3\\
1&&2
\ncline{2,1}{2,3}
```

```
\ncarc[arcangle=-30 ]{2,3}{1,2}
\ncarc[arcangle=-30]{1,2}{2,1}
\nccircle[nodesep=4pt]{->}{2,3}{.6cm}
\endpsmatrix
$
```

2.2 Directions and Weights to Edges

Directions can be shown with the options `{->}`,`{<-}`,`{<->}`. All the times these arrows appears at the end points of the edge. This can be overcome by the package pstricks-add discussed in the section 2.3. Also size of the arrow is fixed but if you want to increase it, see section 2.3.

The distance between arrows and nodes is by default zero. If some separation is required then option `[nodesep=1pt]` is available.

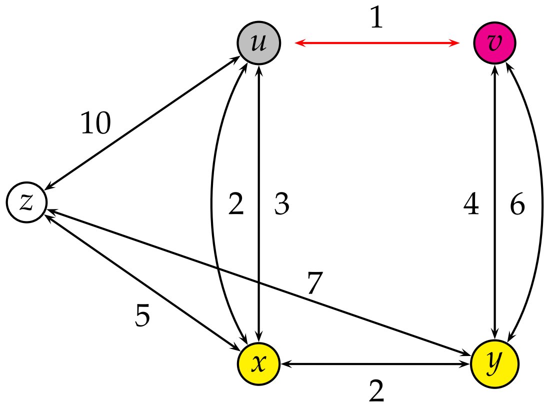
You can add weights to edges. `\ncline{1,1}{1,2}^{1}` will print weight as '1' upside of the edge joining nodes u and v. These options are available:

<	left side of the edge
>	right side of the edge
ˆ	upside of the edge
_	down side of the edge

```
\documentclass[a4paper,12pt]{article}
\usepackage{pstricks,pst-node,pst-tree}
\begin{document}
$
\psmatrix[colsep=2.5cm,rowsep=1.5cm,mnode=circle]
&u&v\\
z\\
&x&y
\ncline[nodesep=5pt]{<->}{1,2}{1,3}^{1}
\ncline{<->}{3,2}{1,2}>{3}
\ncline{<->}{3,2}{3,3}_{2}
\ncarc[arcangle=-30]{<->}{3,3}{1,3}<{6}
\ncarc[arcangle=-30]{<->}{1,2}{3,2}>{2}
\ncline{<-}{2,1}{1,2}<{10}
\ncline{->}{2,1}{3,2}_{5}
```

```
\ncline{<->}{3,3}{2,1}>{7}
\ncline{<->}{1,3}{3,3}<{4}
\endpsmatrix
$
\end{document}
```

With `\ncarc` option 'arcangle' is available. Angle is between 0 and 180 if you go anticlockwise and between 0 and -180 if you go clockwise. If you don't want arrows you can skip the argument `{<->}`.



Look at the red line between u and v. It is separated from nodes. This is the effect of node separation.

2.3 Additional PSTricks

In the above figure observe that arrows are small and at the end of the edge. Some times it may required that arrows should lie in between instead of end of the line. Also, big size arrows, can highlight direction of the edge.

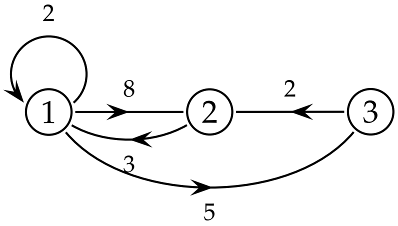
With `pstricks-add` package, we can obtain arrows in between the edge instead of at the end points. Also, one can increase the size of arrows.

$$\boxed{\text{Preamble should include package: pstricks-add}}$$

Make sure that the package is a part of your system, otherwise inclusion of it in the code will cause errors. Download the package from PSTricks site. Include it in your working directory and check that your code is working properly. Then try to save it in the appropriate directory.

```
\documentclass[a4paper,12pt]{article}
\usepackage{pstricks,pst-node,pst-tree,pstricks-add}
\begin{document}
\pagestyle{empty}
$
\psmatrix[colsep=1.5cm,rowsep=1.5cm,mnode=circle]
1&2&3
\psset{arrowscale=2}
\psset{ArrowInside=->,nodesep=1pt}
\everypsbox{\scriptstyle}
\ncline{1,3}{1,2}^{2}
\ncline{1,1}{1,2}^{8}
\ncarc[arcangle=30]{1,2}{1,1}_{3}
\nccircle{->}{1,1}{.5cm}^{2}
\ncarc[arcangle=-50]{1,1}{1,3}_{5}
\endpsmatrix
$
\end{document}
```

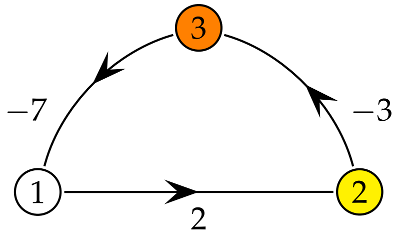
With PSTricks-add we have option `arrowscale=2`. By default arrowscale=1. When we want to change it we can give values like 2, 3 as per our requirement. Normally arrowscale=2 is enough. One more interesting command with pstricks-add is `ArrowInside=->` this will set arrows inside the edges instead of endpoints. That way they are more visible.



```
$
\psmatrix[colsep=1.5cm,rowsep=1.5cm,mnode=circle]
```

```
&[fillstyle=solid,fillcolor=orange]3\\
1&&[fillstyle=solid,fillcolor=yellow]2
\psset{arrowscale=3}
\psset{ArrowInside=->,nodesep=1pt}
\ncline{2,1}{2,3}_{2}
\ncarc[arcangle=-30 ]{2,3}{1,2}>{-3}
\ncarc[arcangle=-30]{1,2}{2,1}<{-7}
\endpsmatrix
$
```

Observe more options with node `[fillstyle=solid, fillcolor=orange]`. This makes the node full of color orange.



3 pstree

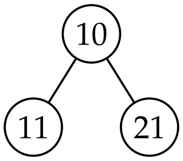
In preamble add one more package `\pstree`. It has tree like structure:
One root and many branches. We can decide what will be the height of the branches. Tree also expands according to the number of branches it has and how healthy branches are i.e. how big is the size of the branch.

With the main command `\pstree{}{}` our tree starts. It's first argument carries root of the tree and second argument is branches if any. Unlike the real tree it grows upside down. The branches of the tree are in pair of braces i.e. in { }. Each branch can be again considered for a new tree to grow. But it should be inside the braces till the whole tree ends.

Heights of the branches are *levels*. It can set by optional argument with pstree as `\pstree[levelsep=35pt]`. It has global effect. You can control level locally by `\pstree[thislevelsep=20pt]`. It has a limited effect for the particular tree.

To obtain circular nodes in pstree use `\Tcircle{10}`. The argument in pair of braces is the matter which you want to print inside the node.

```
\documentclass{article}
\usepackage{graphics,graphicx}
\usepackage{pstricks,pst-node,pst-tree}
\begin{document}
\pstree[levelsep=35pt]{\Tcircle{10}}
{
\Tcircle{11} \Tcircle{21}
}
\end{document}
```



Observe :

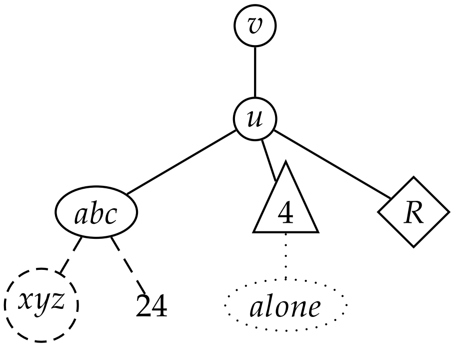
1. pstree does not require $ but you can use $ $ if required.
2. pstree connects nodes i.e. vertices by line segments by default.
3. `levelsep` controls height of the branches.

You can obtain dashed line, dotted lines or 'noline' between two nodes. There are different types of nodes, like triangular, oval shape etc. To get oval use the command `\Toval{abc}`, to obtain diamond shape of the node use `\Tdia{abc}`, to get triangular shape node `\Ttri{abc}` notice the occurrence of capital **T** followed by small **t**, one may skip other 't' thinking that spelling of triangle contains 'tri' so 'Tri' will be fine, **No!**, **T** is for various types of nodes that can be taken as **T**ree nodes, so **T** is necessary.

`\Tr{10}` will produce a node without any shape around 10. Following code illustrates use of various nodes with tree.

```
$\pstree[levelsep=35pt]{\Tcircle{v}}{
\pstree{
\Tcircle{u}}{
\psset{linestyle=solid}\pstree{\Toval{abc}}{
\psset{linestyle=dashed}\Tcircle{xyz} \Tr{24}
}
\pstree{\Ttri{4}}{\psset{linestyle=dotted}\Toval{alone} }\Tdia{R}
}}
$
```

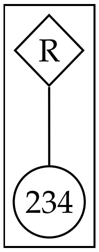
In the code `\psset` command is used. This allows us to set things related to line style, width, arrows etc. in between the code. With linestyle there are four possibilities, none, dashed, dotted, solid. For line thickness use `linewidth=1.5pt`



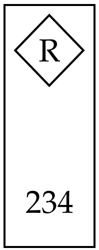
Observe that whatever is line style for the node same is that for the edge joining node from its root. What will happen if we set linestyle as none with node type circle? Let us check.

First draw with linestyle as solid and then with none :

```
\pstree{\Tdia{R}}{\psset{linestyle=solid}\Tcircle{234}}
```



```
\pstree{\Tdia{R}}{\psset{linestyle=none}\Tcircle{234}}
```



As soon as we said `linestyle=none` it removed edge between R and 234 and also drew invisible circle around 234.

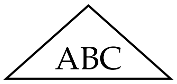
4 Include figures

To include figures, one way is to write the code directly in your running text. `\pstree{\Tdia{R}}{\psset{linestyle=solid}\Tcircle{234}}` after compiling your .tex file in the usual way the figure also gets compiled automatically and will sit in the output at the corresponding place. Yes! you can add code directly with out saying `\includegraphics{}`.

Also, one can include this code in

```
\begin{figure}
 \pstree{\Ttri{ABC}}{}
\end{figure}
```

Which will produce:



Another way is to write the code separately in a file with filename say figno.tex then compile it to get .ps file, from that obtain .eps file with some software like

Gimp or Ghostview then include .eps file just like other .eps files: `\includegraphics{figno.eps}`

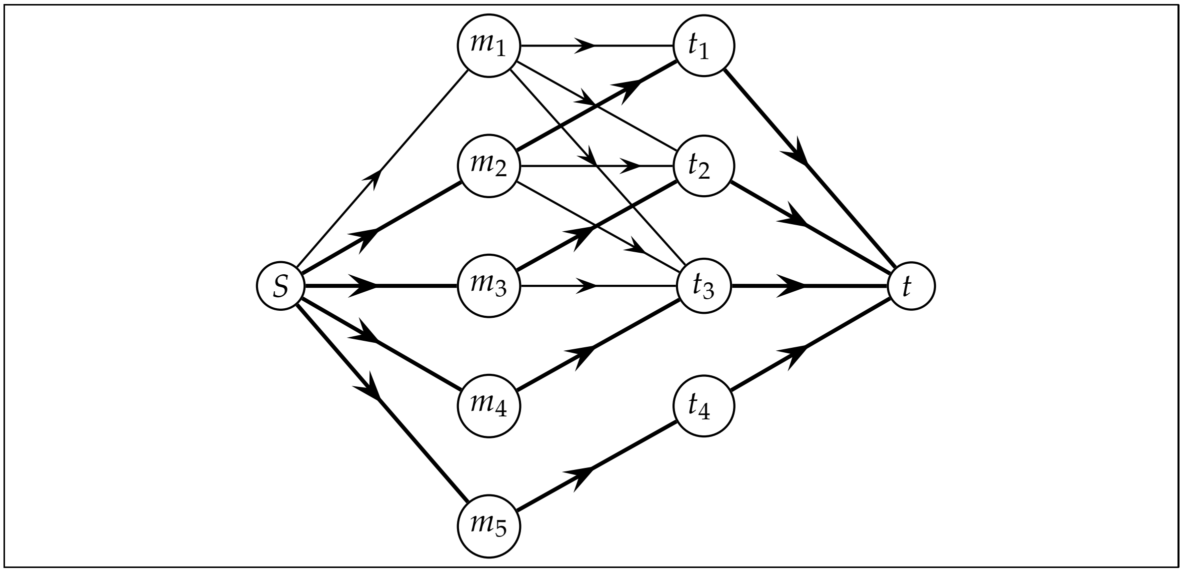
This has some advantage if you want to send your tex file and figures to some one; it is good idea to send it in eps form, so that the receiver gets freedom to adjust the size and place of the figure, and the figure will remain intact. Remember to add `\usepackage{graphics,graphicx}` in the preamble.

Always, convert your file to .ps format. If you want pdf format that can be obtained by `pstopdf` command.

5 Most common mistakes and some tips

- First try some small code of PSTricks in your system to make sure that required packages are already installed in your system.
- Make sure that you have included graphics package in the preamble.
- Use psmatrix inside pair of $'s.
- In psmatrix do not leave blank line; it will cause errors.
- Sometimes we include command `\ncline{2,2}{2,3}` and in the output expect two nodes to get joined but in output we find that there is no line. Check that these 2 nodes are actually present in your matrix. If one of these is not there, how can a line be drawn between nodes one of which is not present? This happens since we can have blank node in between.
- While using pstree environment always write { } braces in pairs and start with new `\pstree` inside these braces. Use some editor like kile (with Linux) which highlights corresponding brackets. It helps a lot.
- Every time write new nested pstree on a new line. That will help to structure your code and it will be easy to edit it.
- After every nested pstree compile the code. If due to some reason your code is giving error and you are not able to fix it, delete or comment to the latest nested pstree and compile again. If the code is error free now. It means the recent pstree has some errors to be corrected. Now again add the code for new nested tree.

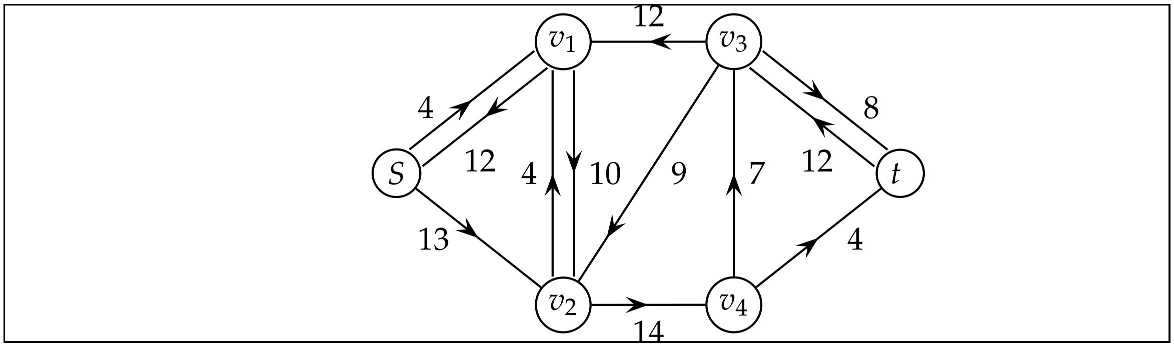
A More figures



• Observe that some lines are thick in the figure. With `\ncline` optional argument `[linewidth=1.5pt]` has been tried. This way we can make a few paths bold to highlight them among others. `linewidth=1pt` is the default line width.

```
$
\psmatrix[colsep=2cm,rowsep=.7cm,mnode=circle]
&m_1&t_1\;\\
&m_2&t_2\;\\
S&m_3&t_3\;&t\;\\
&m_4&t_4\;\\
&m_5&[mnode=none,name=A]
\psset{ArrowInside=->,arrowscale=2}
\ncline{3,1}{1,2}
\ncline[linewidth=1.5pt]{3,1}{2,2}
\ncline[linewidth=1.5pt]{3,1}{3,2}
\ncline[linewidth=1.5pt]{3,1}{4,2}
\ncline[linewidth=1.5pt]{3,1}{5,2}
\ncline[linewidth=1.5pt]{3,2}{2,3}
\ncline{3,2}{3,3}
\ncline[linewidth=1.5pt]{4,2}{3,3}
```

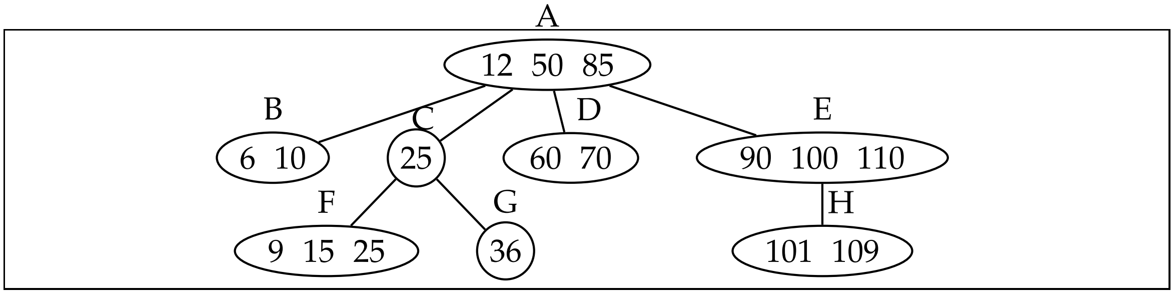
```
\ncline[ArrowInsidePos=.8]{2,2}{3,3}
\ncline[ArrowInsidePos=.8]{2,2}{2,3}
\ncline[linewidth=1.5pt,ArrowInsidePos=.8]{2,2}{1,3}
\ncline{1,2}{3,3}
\ncline{1,2}{2,3}
\ncline{1,2}{1,3}
\ncline[linewidth=1.5pt]{5,2}{4,3}
\ncline[linewidth=1.5pt]{1,3}{3,4}
\ncline[linewidth=1.5pt]{2,3}{3,4}
\ncline[linewidth=1.5pt]{3,3}{3,4}
\ncline[linewidth=1.5pt]{4,3}{3,4}
\endpsmatrix
$
```



• In this figure directions of arrows are different between the same pair of nodes. That can be highlighted with `ArrowInside` option. If two lines between same pair of nodes are drawn they may get overlapped and we cannot point out the opposite directions of lines, also weights of the lines, if any. It is better if we can have small separation between two edges. The option `offset=4pt` is used to achieve the separation.

```
$
\psmatrix[colsep=1.5cm,rowsep=1cm,mnode=circle]
&&v_1&v_3\\
&S &&&t\; \\
&&v_2&v_4
\psset{ArrowInside=->,arrowscale=2}
\ncline{1,4}{1,3}^{12}
```

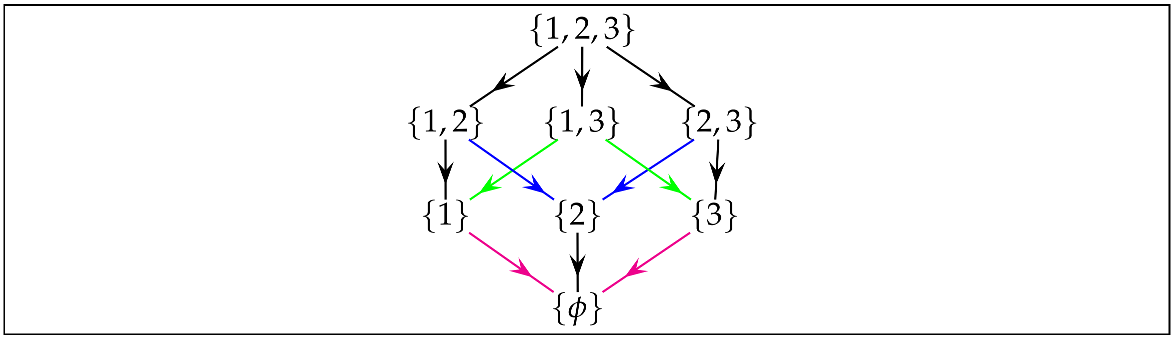
```
\ncline{3,4}{1,4}>{7}
\ncline{3,4}{2,5}>{4}
\ncline{3,3}{3,4}_{14}
\ncline{2,2}{3,3}<{13}
\ncline[ArrowInsidePos=.8]{1,4}{3,3}>[npos=.7]{9}
\psset{nodesep=0pt,offset=4pt}
\ncline{1,3}{3,3}>{10}
\ncline{3,3}{1,3}<{4}
\ncline{1,4}{2,5}>{8}
\ncline{2,5}{1,4}_{12}
\ncline{2,2}{1,3}<{4}
\ncline{1,3}{2,2}_{12}
\endpsmatrix
$
```



• Observe use of `\SpecialCoor` for special coordinates to label object by referring it by its given name. To assign name optional argument with node command can be used. e.g. `\Toval[name=d]{12}`, here 12 will go inside oval shape like 12 and this node labelled by 'd' for internal use. So reader cannot view it.

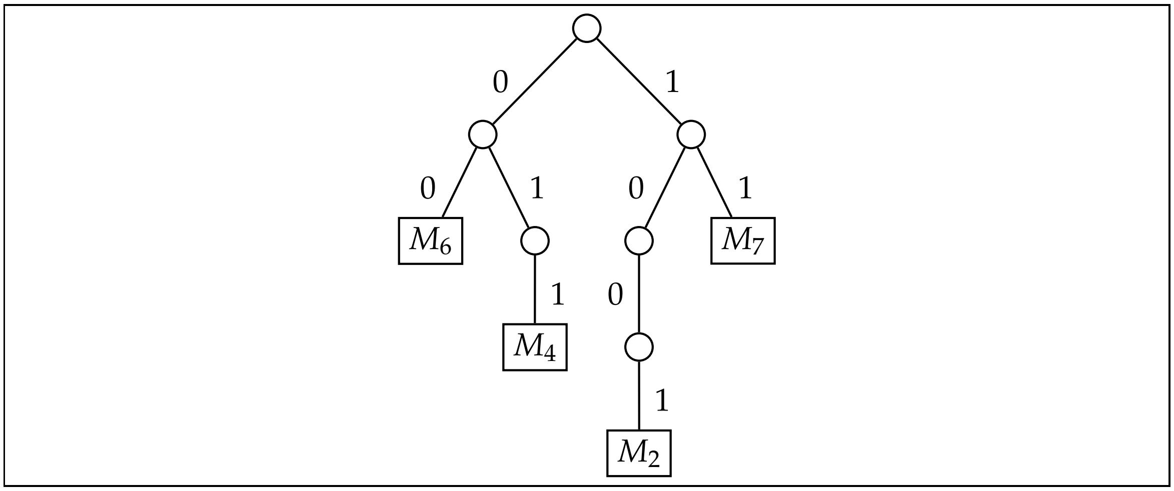
```
\pagestyle{empty}
\SpecialCoor
\pstree[levelsep=35pt,name=a]{\Toval[name=b]{12~~50~~85}}{
\Toval{6~~10}
\pstree{\Tcircle[name=c]{25}}{\Toval[name=f]{9~~15~~25} \Tcircle[name=g]{36}}
\Toval[name=d]{60~~70}
\pstree{\Toval[name=e]{90~~100~~110}}{\Toval[name=h]{101~~109}}}
\uput{.5cm}[90](b){{A}}
\uput{.5cm}[90](a){{B}}
```

```
\uput{.37cm}[80](c){{C}}
\uput{.5cm}[70](d){{D}}
\uput{.5cm}[90](e){{E}}
\uput{.5cm}[90](f){{F}}
\uput{.5cm}[90](g){{G}}
\uput{.5cm}[70](h){{H}}
```



• In the above figure black lines are part of the tree and the line between $\{2,3\}$ and $\{2\}$ is actually line between two nodes from two different *l*evels. To identify nodes, 'name' has been assigned like 'TThree', 'Two' etc. By referring those names, line has been drawn with `\ncline{TThree}{Two}`. Colored lines are lines between nodes of different *levels*.

```
\pstree[levelsep=35pt,ArrowInside=->,ArrowInsidePos=.75,
arrowscale=2,labelsep=14pt]
{\Tr{$\{1,2,3\}$}}{
\pstree{\Tr[name=OT]{$\{1,2\}$}}{
\Tr[name=One]{$\{1\}$}
}
\pstree{\Tr[name=OThree]{$\{1,3\}$}}{
\psset{linestyle=none,ArrowInside=none}{
\pstree{\Tr[name=Two]{$\{2\}$}}
{\psset{linestyle=solid,ArrowInside=->}\Tr[name=Phi]{$\{\phi\}$}}}
}
\pstree{\Tr[name=TThree]{$\{2,3\}$}}{
{\Tr[name=Three]{$\{3\}$}}
\ncline[ArrowInsidePos=.9]{TThree}{Two}
\ncline[ArrowInsidePos=.9]{OThree}{One}
\ncline[ArrowInsidePos=.9]{OThree}{Three}
\ncline{One}{Phi}
\ncline{Three}{Phi}
\ncline[ArrowInsidePos=.9]{OT}{Two}
}
}
```



• To obtain rectangular or square node one possibility is to use `\psframebox{}` which will form box around the argument enclosed in braces. `\psframebox{Example}` will produce $\boxed{\text{Example}}$ in line also.

```
\newcommand{\pbox}[1]{\psframebox{#1}}
$
\pstree[levelsep=40pt]{\Tcircle{~}}{
\pstree[levelsep=40pt]{\Tcircle{~}^{0}}{
\Tr{\pbox{M_6}}^{0}\pstree[levelsep=40pt]{\Tcircle{~}_{1}}{
\Tr{\pbox{M_4}}_{1}}}\pstree[levelsep=40pt]{\Tcircle{~}_{1}}{
\pstree[levelsep=40pt]{\Tcircle{~}^{0}}{
\pstree[levelsep=40pt]{\Tcircle{~}^{0}}{
\Tr{\pbox{M_2}}_{1}}}\Tr{\pbox{M_7}}_{1}
}}
$
```

This is not all. Many commands are not yet discussed, still one can start to get neat figures in Graph Theory with PSTricks. Slowly, as per requirement one can explore new commands through websites. See website [6] for more information on PSTricks. Mailing list of PStricks is additional place to get good support.

References

[1] PSTricks: PostScript macros for Generic TEX, User's Guide, TIMONTHY VAN ZANDT, 1993.

[2] PsTricks-add: additional Macros for pstricks, HERBERT VOSS, 2005.

[3] Drawing Trees with PStricks, DOUG ARNOLD, 2002.

[4] LATEX Companion, MICHEL GOOSSENS, FRANK MITTELBACH, ALEXANDER SAMARIN, Addison-Wesley, 1994.

[5] PsTricks mailing list: pstricks@tug.org

[6] Website http://tug.org/PSTricks/