The **dyntree** package

nsetzer

August 18, 2006

The **dyntree** package is intended for users needing to typeset a dykin tree—a group theoretical construct consisting of cartan coefficients in boxes connected by a series of lines. This package makes it easy for the user to generate these objects by allowing the user to specify only the cartan coefficients and the root number(s) that they connect to below.

This package requires the **coollist** package, which is not a standard **\LaTeX** package but is available at CTAN.

## 1 Basics

To create a Dynkin Tree Diagram, the syntax is as follows:

\begin{verbatim}
\dyntree
\start{dyntree}{⟨num_roots⟩}
\dynbox{⟨cartan_coefficients⟩}{⟨cvs_descendant_root_list⟩} \lend
\finish{dyntree}
\end{verbatim}

where

- ⟨num_roots⟩ is an integer indicating the number of simple roots
- ⟨cartan_coefficients⟩ is an ampersand (&) delimited list of cartan coefficients (the number of which must be equal to ⟨num_roots⟩)
- ⟨cvs_descendant_root_list⟩ is a comma delimited list of integers indicating which simple root can be lowered from this box. The simple roots are numbered from left to right starting at 1 and ending at ⟨num_roots⟩.

Thus, if the group of interest had 3 simple roots, each \texttt{\dynbox} would have a ⟨cartan_coefficients⟩ with three entries; that is, it would be a list with three integers as in

\[1 \& 0 \& 0\]

and the list ⟨cvs_descendant_root_list⟩ would have at most three entries (but it need not have exactly three entries) with the entries being between 1 and 3. The entry 1 would correspond to the first (left-most) simple root being lowered, the entry 2 the second simple root, and 3 the third.

---

*This document corresponds to dyntree v1, dated 2006/08/14.*
So, finally, an entry such as
\begin{verbatim}
\dynbox{-1 & 0 & 0}{1,3}
\end{verbatim}

would specify that the program should draw two lines below the box \[\begin{array}{ccc}
1 & 0 & 0
\end{array}\] ; one for the first simple root, and another for the third simple root. The resulting portion of the diagram would look like

\[\begin{array}{ccc}
-1 & 0 & 0
\end{array}\]

\section*{1.1 Quirks}

There are two quirks with this package, and they are

\begin{itemize}
  \item if there are multiple Dynkin Tree Diagrams required for your document, you must enclose each tree in braces
  \item The lowest state (bottom-most dynbox) must have a non-empty entry in \texttt{\textless cvg\_descendant\_root\_list\textgreater} even though no lines are to be drawn from it. To meet both criteria, place a zero in this spot.
\end{itemize}

\section*{1.2 Examples}

\subsection*{1.2.1 Example 1}

\begin{verbatim}
\{ \start{dyntree}{2} \dynbox{3 & 0}{1} \lend \dynbox{1 & 1}{1,2} \lend \dynbox{2 & -1}{1} \lend \dynbox{-1 & 2}{1,2} \lend \dynbox{0 & 0}{1,2} \lend \dynbox{-3 & 3}{0,2} \lend \dynbox{1 & -2}{1} \lend \dynbox{-2 & 1}{0,2} \lend \dynbox{-1 & -1}{0,2} \lend \dynbox{0 & -3}{0} \lend \finish{dyntree} \}
\end{verbatim}
1.2.2 Example 2

\begin{center}
{\start{dyntree}{3}
\dynbox{2 & 0 & 0}{1} \lend
\dynbox{0 & 1 & 0}{1,2} \lend
\dynbox{1 & -1 & 1}{1,3} \lend
\dynbox{-2 & 2 & 0}{2} \lend
\dynbox{1 & 1 & -1}{1,2} \lend
\dynbox{-1 & 0 & 1}{2,3} \lend
\dynbox{2 & -1 & 0}{1} \lend
\dynbox{-1 & 2 & -1}{2} \lend
\dynbox{0 & -2 & 2}{3} \lend
\dynbox{0 & 0 & 0}{0} \lend
\finish{dyntree}
}
\end{center}
1.2.3 Example 3

This is a $15$ of $SU(5)$:

\begin{center}
\begin{DynTree}
\start{dyntree}{4}
\dynbox{2 & 0 & 0 & 0}{1} \lend
\dynbox{0 & 1 & 0 & 0}{1,2} \lend
\dynbox{1 & -1 & 1 & 0}{1,3}
\dynbox{-2 & 2 & 0 & 0}{2} \lend
\dynbox{1 & 0 & -1 & 0}{1,4}
\dynbox{-1 & 0 & 1 & 0}{2,3} \lend
\dynbox{1 & 0 & 0 &-1}{1}
\dynbox{-1 & 1 & -1 & 1}{2,4}
\dynbox{0 & -2 & 2 & 0}{3} \lend
\dynbox{-1 & 1 & 0 & -1}{2}
\dynbox{0 & -1 & 0 & 1}{3,4} \lend
\dynbox{0 & -1 & 0 & 0 &-1}{3}
\dynbox{0 & 0 & -2 & 2 &0}{4} \lend
\dynbox{0 & 0 & 0 & -1 & 0}{4} \lend
\dynbox{0 & 0 & 0 & -2 &0}{0} \lend
\finish{dyntree}
\end{DynTree}
\end{center}

This is a 15 of $SU(5)$
1.2.4 Example 4

\begin{center}
\{ \\
\begin{dyntree}{2} \\
\dynbox{0 & 2}{2} \lend \\
\dynbox{1 & 0}{1,2} \lend \\
\dynbox{2 & -2}{1} \\
\dynbox{-1 & 1}{2} \lend \\
\dynbox{0 & -1}{1} \lend \\
\dynbox{-2 & 0}{0} \lend \\
\finish{dyntree} \\
\end{dyntree}
\}
\end{center}
2 Implementation

2.1 Variables and Constants

The dynree package utilizes a picture environment to create the tree. To do this it requires several constant length values, as well as calculated length values and counters. These are define below.

2.1.1 Cartan Coefficients Box Variables

The first thing to do is declare the length variables associated with a cartan coefficients box—dynbox for short. These variables are

\begin{itemize}
  \item \DYNTREE@widechar the width of a \texttt{-1}
  \item \DYNTREE@thinchar the width of a \texttt{1}
  \item \DYNTREE@cartancoefwidth the width of a cartan coefficient (a combination of \texttt{-1} and \texttt{1})
  \item \DYNTREE@marginwidth the width of the margin of the cartan coefficients box
  \item \DYNTREE@colsepwidth the width between columns of the cartan coefficients box
  \item \DYNTREE@dyboxheight the height (baseline to top) of the cartan coefficients box
  \item \DYNTREE@dyboxdepth the depth (baseline to bottom) of the cartan coefficients box
  \item \DYNTREE@dyboxvlen the full vertical height of a cartan coefficients box
  \item \DYNTREE@dyboxwidth the width of a dynbox (calculated based on numroots)
\end{itemize}

\newlength{\DYNTREE@widechar}%
2 \newlength{\DYNTREE@thinchar} \\
3 \newlength{\DYNTREE@cartancoefwidth} \\
4 \newlength{\DYNTREE@marginwidth} \\
5 \newlength{\DYNTREE@colsepwidth} \\
6 \newlength{\DYNTREE@dynboxheight} \\
7 \newlength{\DYNTREE@dynboxdepth} \\
8 \newlength{\DYNTREE@dynboxvlen} \\
9 \newlength{\DYNTREE@dynboxwidth} \\

Now that they are declared, initialize the “exterior” ones

10 \settowidth{\DYNTREE@widechar}{$-1$} \\
11 \settowidth{\DYNTREE@thinchar}{$1$} \\
12 \setlength{\DYNTREE@cartancoefwidth}{}{\DYNTREE@widechar*1/2 + \DYNTREE@thinchar*1/2} \\
13 \settowidth{\DYNTREE@marginwidth}{}{$\begin{array}{|c|} \hline 1 \end{array}$} \\
14 \settowidth{\DYNTREE@colsepwidth}{}{$\begin{array}{|cc|} \hline 1 & 1 \end{array}$} \\
15 \addtolength{\DYNTREE@marginwidth}{}{-\DYNTREE@thinchar} \\
16 \settowidth{\DYNTREE@colsepwidth}{}{$\begin{array}{|cc|} \hline 1 & 1 \end{array}$} \\
17 \addtolength{\DYNTREE@marginwidth}{}{-\DYNTREE@thinchar*2} \\
18 \settoheight{\DYNTREE@dynboxheight}{}{$\begin{array}{|c|} \hline 1 \end{array}$} \\
19 \settodepth{\DYNTREE@dynboxdepth}{}{$\begin{array}{|c|} \hline 1 \end{array}$} \\
20 \setlength{\DYNTREE@dynboxvlen}{}{\DYNTREE@dynboxheight + \DYNTREE@dynboxdepth} \\

and now for convenience and error testing, print them out

widechar: 12.77782pt  \\
thinchar: 5.00002pt  \\
cartancoefwidth: 8.8889pt  \\
marginwidth: 10.0pt  \\
colsepwidth: 10.0pt  \\
dynboxheight: 8.9pt  \\
dynboxdepth: 3.9pt  \\
dynboxvlen: 12.79999pt  \\

\subsection{Dynkin Tree Variables}

These variables are specific to the actual creation of the tree structure.
counters
DYNTREE@numlevel  the number of levels in the tree
DYNTREE@nextlevel the number of the next level
DYNTREE@numboxes counter for counting number of boxes in a row
DYNTREE@nextnumboxes counter for counting number of boxes in the next row
DYNTREE@target used to record the ‘targeted’ array element when sorting
DYNTREE@listlen the length of the list of descendents
DYNTREE@xCoord  the x coordinate in scaled points
DYNTREE@yCoord  the y coordinate in scaled points
DYNTREE@xPos  the x coordinate in scaled points
DYNTREE@yPos  the y coordinate in scaled points
DYNTREE@xComp  the x coordinate in scaled points
DYNTREE@yComp  the y coordinate in scaled points
DYNTREE@leftX  the left most x coordinate in scaled points
DYNTREE@ct  generic counter
DYNTREE@counter  generic counter
DYNTREE@index  generic counter
DYNTREE@root  the root number

lengths
\DYNTREE@dynboxsep  the distance between dynkin boxes
\DYNTREE@levelsep  the distances between each level (from dynkin box bottom to top of next layer’s dynbox)
\DYNTREE@leftmostX  the left most x value
\DYNTREE@rightmostX  the right most x value
\DYNTREE@unitlen  the unit length value before altering (to allow it to be restored)
\DYNTREE@templen  temporary length storage
\DYNTREE@holdlen  temporary length storage

commands
\DYNTREE@treestop indicates the point where the gobbler stops reading (to allow \LaTeX to properly read all the data. Its value is \&\&\&\&
\DYNTREE@treeend indicates the end of the tree. It is in the definition of the gobbler. It has a value of \%\%\%\%

\lend  And there are two external commands:
\lend  indicates the end of one tree level. The value is never used: this token is used as a delimiter by the user and the code. Its value, which should never be typed, is \&\&\&\&\%
\lend  indicates the start of a dynbox. The value is never used: this token is purely for delimiting the start of the dynbox by the user; it is only defined to satisfy the ifthenelse statement

25 \newcounter{DYNTREE@numlevel}
2.2 The Tree Eater

While the Dynkin tree structure resembles that of an environment, it actually consists of a \texttt{Dynkin} command that consumes the data, sorts it, and displays the proper items. The basis of this consumption are several “gobblers”—the first of which eats the tree one level at a time

\begin{verbatim}
\def\DYNTREE@gobbletree#1\lend#2\DYNTREE@treeend{% Before the level can be processed, several things must be adjusted. First, since a new level is beginning the counter must be incremented by one

\setcounter{DYNTREE@numlevel}{1}%
\setcounter{DYNTREE@numboxes}{0}%
\setcounter{DYNTREE@nextnumboxes}{0}%
\newcounter{DYNTREE@nextlevel}%
\newcounter{DYNTREE@numboxes}%
\newcounter{DYNTREE@nextnumboxes}%
\newcounter{DYNTREE@target}%
\newcounter{DYNTREE@listlen}%
\newcounter{DYNTREE@xCoord}%
\newcounter{DYNTREE@yCoord}%
\newcounter{DYNTREE@xPos}%
\newcounter{DYNTREE@yPos}%
\newcounter{DYNTREE@xComp}%
\newcounter{DYNTREE@yComp}%
\newcounter{DYNTREE@leftX}%
\newcounter{DYNTREE@ct}%
\newcounter{DYNTREE@counter}%
\newcounter{DYNTREE@index}%
\newcounter{DYNTREE@root}%
\newlength{\DYNTREE@dynboxsep}%
\newlength{\DYNTREE@levelsep}%
\newlength{\DYNTREE@leftmostX}%
\newlength{\DYNTREE@unitlen}%
\newlength{\DYNTREE@templen}%
\newlength{\DYNTREE@holdlen}%
\newcommand{\DYNTREE@treestop}{\&\&\&}%
\newcommand{\DYNTREE@treeend}{\%\%\%}%
\newcommand{\lend}{\&\%\%}%
\newcommand{\dynbox}{}%
\end{verbatim}

Now that they are declared, initialize the “exterior” ones

\setlength{\DYNTREE@dynboxsep}{\DYNTREE@colsepwidth}%
\setlength{\DYNTREE@levelsep}{1cm}%
\setlength{\DYNTREE@colsepwidth}{1cm}%
\setlength{\DYNTREE@colsepwidth}{1cm}%

2.2 The Tree Eater

While the Dynkin tree structure resembles that of an environment, it actually consists of a \texttt{Dynkin} command that consumes the data, sorts it, and displays the proper items. The basis of this consumption are several “gobblers”—the first of which eats the tree one level at a time

\begin{verbatim}
\def\DYNTREE@gobbletree#1\lend#2\DYNTREE@treeend{% Before the level can be processed, several things must be adjusted. First, since a new level is beginning the counter must be incremented by one

\setcounter{DYNTREE@numlevel}{1}%
\setcounter{DYNTREE@numboxes}{0}%
\setcounter{DYNTREE@nextnumboxes}{0}%
\newcounter{DYNTREE@nextlevel}%
\newcounter{DYNTREE@numboxes}%
\newcounter{DYNTREE@nextnumboxes}%
\newcounter{DYNTREE@target}%
\newcounter{DYNTREE@listlen}%
\newcounter{DYNTREE@xCoord}%
\newcounter{DYNTREE@yCoord}%
\newcounter{DYNTREE@xPos}%
\newcounter{DYNTREE@yPos}%
\newcounter{DYNTREE@xComp}%
\newcounter{DYNTREE@yComp}%
\newcounter{DYNTREE@leftX}%
\newcounter{DYNTREE@ct}%
\newcounter{DYNTREE@counter}%
\newcounter{DYNTREE@index}%
\newcounter{DYNTREE@root}%
\newlength{\DYNTREE@dynboxsep}%
\newlength{\DYNTREE@levelsep}%
\newlength{\DYNTREE@leftmostX}%
\newlength{\DYNTREE@unitlen}%
\newlength{\DYNTREE@templen}%
\newlength{\DYNTREE@holdlen}%
\newcommand{\DYNTREE@treestop}{\&\&\&}%
\newcommand{\DYNTREE@treeend}{\%\%\%}%
\newcommand{\lend}{\&\%\%}%
\newcommand{\dynbox}{}%
\end{verbatim}

Now that they are declared, initialize the “exterior” ones

\setlength{\DYNTREE@dynboxsep}{\DYNTREE@colsepwidth}%
\setlength{\DYNTREE@levelsep}{1cm}%
\setlength{\DYNTREE@colsepwidth}{1cm}%
\setlength{\DYNTREE@colsepwidth}{1cm}%
Process the level:
59 \DYNTREE@gobbledynboxes#1\lend%

Record the number of boxes for this level
60 \expandafter\xdef%
61 \csname DYNTREE@level@\roman{DYNTREE@numlevel}@numbox\endcsname%
62 {\arabic{DYNTREE@numboxes}}%

Now check for the signal to stop processing
63 \ifthenelse{ \equal{#2}{\DYNTREE@treestop} }%
64 {%

The End—just do nothing
65 %}
66 % Else
67 {%

continue processing levels until the end of the tree
68 \DYNTREE@gobbletree#2\DYNTREE@treeend%
69 %}
70 %

2.3 The Box Eater

The second “gobbler” eats the boxes one by one:
71 \def\DYNTREE@gobbledynboxes#1\dynbox#2#3#4{%

increment the number of boxes
72 \addtocounter{DYNTREE@numboxes}{1}%

Store the boxes for this level in registers
73 \expandafter\newbox%
74 \csname DYNTREE@box@\roman{DYNTREE@numlevel}@\roman{DYNTREE@numboxes}\endcsname%
75 \expandafter\setbox%
76 \csname DYNTREE@box@\roman{DYNTREE@numlevel}@\roman{DYNTREE@numboxes}\endcsname=%
77 \hbox{$\begin{array}{|*{\DYNTREE@numroots}{c}|} \hline #2 \ \hline \end{array}$}%

Calculate the X value for each descendent and place in sorted list

* Get the length of the list
78 \listlenstore{DYNTREE@listlen}{#3}%
79 \expandafter\xdef
80 \csname DYNTREE@childline@\roman{DYNTREE@numlevel}@\roman{DYNTREE@numboxes}@boxnum\endcsname%
81 {\arabic{DYNTREE@listlen}}%
82 \ifthenelse{\value{DYNTREE@listlen} > \DYNTREE@numroots}%
83 {%
84 \PackageError{dyntree}%
85 {}% Length of descendant of \arabic{DYNTREE@numboxes}%
86 on level \arabic{DYNTREE@numlevel} exceeds number of roots%
87 \DYNTREE@numroots)%
88 %}
store the list in a temp variable for convenience in typing, store it more permanently for use later on.

\liststore{#3}{DYNTREE@templist@}
\liststore{#3}
\liststore{#3}{DYNTREE@childlist@}\roman{DYNTREE@numlevel}@\roman{DYNTREE@numboxes}@%

* go through the list and generate the sorted 'array' DYNTREE@level@\(level@\)(boxnum)@X

\setcounter{DYNTREE@ct}{1}
\whiledo{ \NOT \(\value{DYNTREE@ct} > \value{DYNTREE@listlen}\) \AND
\NOT\(\value{DYNTREE@ct} > \DYNTREE@numroots\) }{

need to store the ‘root’ value in a counter to retrieve the data
\setcounter{DYNTREE@counter}{\csname DYNTREE@templist@\roman{DYNTREE@ct}\endcsname}%

check that the number submitted is within the allowed range
\ifthenelse{ \(\value{DYNTREE@counter} > \DYNTREE@numroots\) \OR
\value{DYNTREE@counter} < 1\) }{
\ifthenelse{\value{DYNTREE@counter} = 0}{}{
\PackageError{dyntree}{Descendant root of \arabic{DYNTREE@numboxes} on level\arabic{DYNTREE@numlevel} out of bounds\(\arabic{DYNTREE@counter} > \DYNTREE@numroots\)}%}
}
}

Do nothing - this is the last level

\PackageError{dyntree}%
}\%
\%
\%

temporarily store the length
\setlength{\DYNTREE@templen}%
\addtolength{\DYNTREE@templen}{\csname DYNTREE@rootX@\roman{DYNTREE@counter}\endcsname*(-1)}%

adjust the left most length
\ifthenelse{\DYNTREE@templen < \DYNTREE@leftmostX}{}{
\setlength{\DYNTREE@leftmostX}{\DYNTREE@templen}}%
\begin{verbatim}
127 % Else
128 {\%}
129 \setlength{\DYNTREE@holdlen}{\DYNTREE@templen}\%
130 \addtolength{\DYNTREE@holdlen}{\DYNTREE@dynboxwidth}\%
131 \ifthenelse{ \DYNTREE@holdlen > \DYNTREE@rightmostX }\%
132 {\%}
133 \setlength{\DYNTREE@rightmostX}{\DYNTREE@holdlen}\%
134 }\%
135 % Else
136 {\%}
137 \setlength{\DYNTREE@holdlen}\%
138 \expandafter\csname DYNTREE@dynboxX@roman{DYNTREE@counter}@endcsname\%
139 \addtolength{\DYNTREE@holdlen}{\DYNTREE@templen}\%
140 \setcounter{DYNTREE@xPos}{\DYNTREE@holdlen}\%

counter has served its purpose and may be used in another context
141 \setcounter{DYNTREE@counter}{\value{DYNTREE@nextnumboxes}}\%
* initialize the counter to the END of the array
142 \setcounter{DYNTREE@nextlevel}{\value{DYNTREE@numlevel} + 1}\%
* Check for array elements
143 \ifthenelse{\value{DYNTREE@counter} = 0}\%
144 {\%
145 \edef\DYNTREE@temparray{\csname DYNTREE@level@roman{DYNTREE@nextlevel}@roman{DYNTREE@counter}@X\endcsname}\%
146 \edef\DYNTREE@tempinsert{\the\DYNTREE@templen}\%
147 \whiledo{ \value{DYNTREE@counter} > 0 \AND \lengthtest{\DYNTREE@tempinsert < \DYNTREE@temparray} }\%
148 {\%
149 \edef\DYNTREE@temparray{\csname DYNTREE@level@roman{DYNTREE@nextlevel}@roman{DYNTREE@counter}@X\endcsname}\%
150 }\%
151 }\%
152 \edef\DYNTREE@temparray\%
153 \expandafter\csname DYNTREE@level@roman{DYNTREE@nextlevel}@roman{DYNTREE@counter}@0\endcsname\%
154 \edef\DYNTREE@tempinsert{\the\DYNTREE@templen}\%
155 \find where element should be inserted
156 \whiledo{ \value{DYNTREE@counter} > 0 \AND \lengthtest{\DYNTREE@tempinsert < \DYNTREE@temparray} }\%
157 {\%
158 \edef\DYNTREE@temparray\%
159 \csname DYNTREE@level@roman{DYNTREE@nextlevel}@roman{DYNTREE@counter}@0\endcsname\%
\end{verbatim}
the thing needs to be inserted at
if they aren't equal, move from target to nextnumboxes up to target+1 to
nextnumboxes+1

ifthenelse( \NOT \lengthtest{\DYNTREE@tempinsert = \DYNTREE@temparray} )
{%
\setcounter{DYNTREE@nextnumboxes}{\value{DYNTREE@counter} + 1}%
\whiledo{ \value{DYNTREE@counter} > \value{DYNTREE@target} }%
{%
get the value in the array spot one before
\addtocounter{DYNTREE@counter}{-1}%
edef\DYNTREE@tempswap%
\csname DYNTREE@level@roman{DYNTREE@nextlevel}@roman{DYNTREE@counter}@X\endcsname%
store this value in the next array spot
\addtocounter{DYNTREE@counter}{1}%
\expandafter\xdef\csname DYNTREE@level@roman{DYNTREE@nextlevel}@roman{DYNTREE@counter}@X\endcsname%
\DYNTREE@tempswap%
\addtocounter{DYNTREE@counter}{-1}%
}%
% Else
{%
Do nothing
}%
%
\ifthenelse{\equal{#4}{\lend}}%
{%
\addtocounter{DYNTREE@ct}{1}%
%}
% Else
{%

%
2.4 The Dyntree Environment

Dynkin Tree Environment

```
\def\start#1#2#3\finish#4%
{\
  \ifthenelse{\equal{#1}{#4} \AND \equal{#4}{dyntree}}{
    \providecommand{\DYNTREE@numroots}{#2}
  }{
    \providecommand{\DYNTREE@numroots}{#2}
  }
}
```

Initialize Interior Dynkin Box Variables

```
\setlength{\DYNTREE@dynboxwidth}{\DYNTREE@marginwidth + \DYNTREE@cartancoefwidth*\DYNTREE@numroots + \DYNTREE@colsepwidth*(\DYNTREE@numroots-1)}
```

Initialize Interior Dynkin Tree Variables

```
\setlength{\DYNTREE@leftmostX}{0pt}
\setlength{\DYNTREE@rightmostX}{\DYNTREE@dynboxwidth}
```

the highest left starts at zero, so initialize this value

```
\expandafter\gdef\csname DYNTREE@level@i@i@X\endcsname{0pt}
```

There are no levels, so initialize numlevel to zero

```
\setcounter{DYNTREE@numlevel}{0}
```

Determine the root lines and dynkin box offsets

```
\setcounter{DYNTREE@ct}{1}
\whiledo{\NOT \value{DYNTREE@ct}>\DYNTREE@numroots}}{
\setlength{\DYNTREE@templen}{(\DYNTREE@dynboxwidth + \DYNTREE@dynboxsep)*\value{DYNTREE@ct} - (\DYNTREE@dynboxwidth + \DYNTREE@dynboxsep)*1/2 - (\DYNTREE@dynboxwidth + \DYNTREE@dynboxsep)*\DYNTREE@numroots/2}
\expandafter\xdef\csname DYNTREE@rootX@\roman{DYNTREE@ct}\endcsname{\the\DYNTREE@templen}
\setlength{\DYNTREE@rootX@\roman{DYNTREE@ct}}{\value{DYNTREE@ct} + \DYNTREE@numlevel\DYNTREE@cartancoefwidth}
```

store the actual length in a command

```
\expandafter\xdef\csname DYNTREE@rootX@\roman{DYNTREE@ct}\endcsname{\the\DYNTREE@rootX@\roman{DYNTREE@ct}}
```

Calculate the dynkin box x offset and store it in a temporary length

```
\setlength{\DYNTREE@templen}{\DYNTREE@dynboxwidth + \DYNTREE@dynboxsep}
\setlength{\DYNTREE@templen}{\DYNTREE@templen + \DYNTREE@numlevel\DYNTREE@cartancoefwidth}
```

store the actual length in a command

```
\expandafter\xdef\csname DYNTREE@rootX@\roman{DYNTREE@ct}\endcsname{\the\DYNTREE@rootX@\roman{DYNTREE@ct}}
```

Calculate the dynkin box x offset and store it in a temporary length

```
\setlength{\DYNTREE@templen}{\DYNTREE@tempX}
```

Eat the boxes until there are no more
\begin{verbatim}
227 \DYNTREE@dynboxwidth/2/\DYNTREE@numroots +
228 \DYNTREE@dynboxwidth*(\value{DYNTREE@ct}-1)/\DYNTREE@numroots
229 \}%
230 \expandafter\def\csname DYNTREE@dynboxX\roman{DYNTREE@ct}\endcsname%
231 {\the\DYNTREE@templen}%
232 \addtocounter{DYNTREE@ct}{1}%
233 \}%
234 \DYNTREE@gobbletree#3\DYNTREE@treestop\DYNTREE@treeend%
235 %
236 %
237 %
238 %
239 %
240 %
241 \setlength{\DYNTREE@unitlen}{\unitlength}%
242 \setlength{\unitlength}{1sp}%
243 %
244 %
245 \setcounter{DYNTREE@xCoord}{\DYNTREE@rightmostX - \DYNTREE@leftmostX}%
246 %
247 \setcounter{DYNTREE@leftX}{\DYNTREE@leftmostX}%
248 %
249 %
250 \begin{picture}(\arabic{DYNTREE@xCoord},\arabic{DYNTREE@yCoord})%
251 (\value{DYNTREE@leftX},0)%
252 \setcounter{DYNTREE@ct}{1}%
253 \whiledo{\NOT \( \value{DYNTREE@ct} > \value{DYNTREE@numlevel} \)}%
254 {%
255 \setcounter{DYNTREE@counter}{1}%
256 \setlength{\DYNTREE@templen}%;
257 \def\DYNTREE@tempboxnum{\csname DYNTREE@level\roman{DYNTREE@ct}@numbox\endcsname}%
258 \whiledo{\NOT \( \value{DYNTREE@counter} > \DYNTREE@tempboxnum \)}%
259 {%
260 \end{verbatim}

15
grab the value of the x coordinate (it’s a length but not stored as one)
\xdef\DYNTREE@tempxCoord%
\expandafter\csname DYNTREE@level@\roman{DYNTREE@ct}@\roman{DYNTREE@counter}@X@endcsname%
% convert x coordinate to a length
\settlength{\DYNTREE@holdlen}{\DYNTREE@tempxCoord}%
% convert length to an integer in scaled points (sp)
\setcounter{DYNTREE@xCoord}{\DYNTREE@holdlen}%
% place each dynkin box
\put(\arabic{DYNTREE@xCoord},\arabic{DYNTREE@yCoord}){
\expandafter\copy\csname DYNTREE@box@\roman{DYNTREE@ct}@\roman{DYNTREE@counter}\endcsname%
}% go through the descendants and place the lines
\setcounter{DYNTREE@listlen}%%
\expandafter\csname DYNTREE@childline@\roman{DYNTREE@ct}@\roman{DYNTREE@counter}@boxnum\endcsname%
\setcounter{DYNTREE@index}{1}%%
\whiledo{\NOT \value{DYNTREE@index} > \value{DYNTREE@listlen}}%%
\xdef\DYNTREE@childroot%%
\ifthenelse{\NOT \DYNTREE@childroot = 0 }%%
\setcounter{DYNTREE@root}{\DYNTREE@childroot}%%
\setcounter{DYNTREE@xPos}{\value{DYNTREE@xCoord}}%%
\setcounter{DYNTREE@xComp}{\value{DYNTREE@xCoord}}%%
\setcounter{DYNTREE@yPos}{\value{DYNTREE@yCoord}}%%
\setcounter{DYNTREE@yComp}{\value{DYNTREE@yCoord}}%%
\xdef\DYNTREE@temprootX%%
\settlength{\DYNTREE@templen}{\DYNTREE@temprootX}%%
\addtocounter{DYNTREE@xPos}{\DYNTREE@templen*(-1)}%%
\xdef\DYNTREE@dyoffset%%
\settlength{\DYNTREE@templen}{\DYNTREE@dyoffset}%%
\addtocounter{DYNTREE@xPos}{\DYNTREE@templen*(-1)}%%
\xdef\DYNTREE@dyoffset%%
\settlength{\DYNTREE@templen}{\DYNTREE@dyoffset}%%
\addtocounter{DYNTREE@xPos}{\DYNTREE@templen}%%
\addtocounter{DYNTREE@xComp}{\DYNTREE@templen}%%
\settlength{\DYNTREE@templen}{1pt} for frame thickness
Change History

v1.0
  General: Initial Release . . . . . 1

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the entry is used.

<table>
<thead>
<tr>
<th>Symbols</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>%</td>
<td>\dynbox . . . . . . . . . . . $I$, 25, 52, 71</td>
</tr>
<tr>
<td>&amp;</td>
<td>\dyntree (environment) . . . . . . $I$</td>
</tr>
<tr>
<td>&amp;</td>
<td>\DYNTREE@cartancoefwidth . . 3, 12, 207</td>
</tr>
<tr>
<td></td>
<td>\DYNTREE@childroot . . . . . 283, 288, 290</td>
</tr>
</tbody>
</table>