Abstract

This is a \LaTeX{} package that provides Ti\kZ{}-based macros to draw game trees. The main idea underlying the core macros here is the completion of a whole tree by using a sequence of simple 'parent-child' tree structures, with no longer nested relations involved like the use of grandchildren or great-grandchildren. With the \texttt{istgame} package, you can draw game trees as easily as drawing game trees with pen and paper.

\textbf{Keywords:} game trees, nodes, branches, information sets, continuum of branches, subgames

Table of Contents

0 Changes and remarks .................................................. 1
  0.1 Changes in version 2.1 ........................................... 1
  0.2 Changes in version 2.0 ........................................... 3
  0.3 What’s new in version 2.0 ...................................... 4
  0.4 How to read this document ...................................... 5
  0.5 Remarks .............................................................. 6
  0.6 Previous changes (up to version 1.0) ......................... 7
1 Getting started .......................................................... 8
  1.1 Getting-started example: a simple tree ...................... 8
  1.2 Connecting simple tree structures ............................ 8
  1.3 Complete examples for desperate users ....................... 9
    1.3.1 How to put a decision node and its owner .............. 9
    1.3.2 How to print branches with action labels and payoffs . 9
    1.3.3 How to put information sets .............................. 10
    1.3.4 How to put a continuum of branches .................... 11
    1.3.5 How to change the growing direction of a tree ........ 11
2 Important distances: \texttt{\textbackslash xtdistance} .............. 12
3 The \texttt{istgame} environment and node styles ................ 13
  3.1 The \texttt{istgame} environment ................................ 13
  3.2 Node styles ....................................................... 13
    3.2.1 Basic node styles ........................................ 13
    3.2.2 Your own node styles: \texttt{\textbackslash setistNewNodeStyle} . 14
4 Core macro: \texttt{\textbackslash istroot} ............................ 16
  4.1 \texttt{\textbackslash istroot}: basics ............................ 16
    4.1.1 \texttt{\textbackslash istroot} – counterclockwise: standard version . 16
    4.1.2 \texttt{\textbackslash istroot'} – clockwise: swap version .............. 18
  4.2 \texttt{\textbackslash istrooto}: oval version ..................... 19
    4.2.1 \texttt{\textbackslash istrooto} – counterclockwise .................. 19
    4.2.2 \texttt{\textbackslash istrooto'} – clockwise: swap version .......... 20
5 Core macro: \texttt{\textbackslash istb} .......................... 21
10 Information sets ........................................ 49
 10.1 \xtInfoset: standard version .......................... 49
 10.2 \xtInfosetO: oval version ............................. 50
    10.2.1 \xtInfosetO: basics .............................. 50
    10.2.2 Sloped information sets ......................... 52

5.1 \istb: basics ........................................... 21
 5.1.1 Basics: branches, action labels, and payoffs ........ 21
 5.1.2 Printing action labels in italics in text mode: \setistmathTF* .......... 24

5.2 \istb*: starred version ................................. 25
 5.2.1 \istb*: basics ..................................... 25
 5.2.2 \xtShowEndPoints and \xtHideEndPoints .................. 25

5.3 \istbt: terminal version ................................ 26
 5.3.1 \istbt: basics ..................................... 26
 5.3.2 \xtShowTerminalNodes and \xtHideTerminalNodes ................... 26

5.4 \istb: dual action label version ....................... 27
 5.4.1 \istb: basics ..................................... 27
 5.4.2 \xtActionLabel and \xtActionLabel* ..................... 28

5.5 \istbA: alternative (or arc) version ................. 29
 5.5.1 \istbA: basics ..................................... 29
 5.5.2 \istbA: applications ................................. 29

6 Important labels: players, action labels, and payoffs ........ 30
 6.1 How to put players .................................... 30
    6.1.1 Players: basics .................................. 30
    6.1.2 Coloring players or a whole simple tree using \istroot ............. 30
    6.1.3 Decorating players or a whole simple tree using \istrooto .......... 31
    6.1.4 Changing the input mode and text font style: \setistmathTF(*) .... 32

6.2 How to put action labels ................................ 33
    6.2.1 Action labels: basics ................................ 33
    6.2.2 Decorating action labels ................................ 34
    6.2.3 Sloped action labels ................................ 34
    6.2.4 Dual action labels .................................. 35
    6.2.5 Changing the input mode and text font style: \setistmathTF(*) .... 35

6.3 How to put payoffs ..................................... 36
    6.3.1 Payoffs: basics ................................... 36
    6.3.2 Payoffs and south .................................. 37
    6.3.3 Decorating payoffs .................................. 38
    6.3.4 Changing the input mode and text font style: \setistmathTF(*) .... 39

7 Input mode and text font style changer: \setistmathTF(*) ........ 39
  7.1 \setistmathTF: input mode changer ..................... 40
  7.2 \setistmathTF*: input mode and text font style changer ............ 41
  7.3 \setistmathTF* and supplementary macros \xtFoo* printing labels .... 41

8 Fine-tuning positions of players, action labels, and payoffs (experimental) ........ 43
  8.1 Fine-tuning positions: owners ............................... 43
  8.2 Fine-tuning positions: action labels ....................... 43
    8.2.1 Abbreviations: [1], [r], [a], and [b] .......................... 43
    8.2.2 Abbreviations: [al], [ar], [bl], and [br] ......................... 44
  8.3 Fine-tuning positions: payoffs .............................. 45

9 Growing direction of trees .................................. 45
  9.1 \isetstgrodirection – counterclockwise .................... 46
  9.2 \isetstgrodirection – clockwise .......................... 46
  9.3 Examples of rotating trees with \isetstgrodirection ............ 47
    9.3.1 A tree growing east – counterclockwise ..................... 47
    9.3.2 A tree growing east – clockwise .......................... 47
    9.3.3 A tree growing north – counterclockwise ................... 48
    9.3.4 A tree growing north – clockwise .......................... 48
    9.3.5 \isetxtgrodkey for one simple tree .......................... 48

10 Information sets ........................................ 49
10.2.3 \setxtinfofstyle ........................................ 52
10.2.4 \setxtinfosetlayer .................................. 53
10.3 \xtCInfoset: curved version ................................. 54
  10.3.1 Curved information sets with \xtCInfoset: basics .......... 54
  10.3.2 Skewed \xtCInfoset: full functions .................... 54
10.4 \xtCInfosetO: curved oval version .......................... 56
  10.4.1 Curved bubble type information sets with \xtCInfosetO: basics ... 56
  10.4.2 Skewed \xtCInfosetO: full function ..................... 57
10.5 Fine-tuning \xtCInfosetO: Not for most users ................ 57
  10.5.1 \xtCInfosetOTurnX: too high or too low midpoint factor .... 57
  10.5.2 Scaling information sets according to the value of TikZ scale .... 59
11 Continuum of branches ........................................ 60
  11.1 \istrootcntm: standard continuum version ................. 60
    11.1.1 \istrootcntm: basics ................................ 60
    11.1.2 \cntmdistance ...................................... 62
    11.1.3 \cntmdistance*: combined with \xtdistance ............... 63
    11.1.4 \cntmpreset: controlling continuum triangles ............ 63
    11.1.5 \cntmpreset*: simple triangles with no background color .... 64
  11.2 \istrootcntmA: continuum arc version ..................... 65
    11.2.1 \istrootcntmA: basics ................................ 65
    11.2.2 \istrootcntmA and \istbA .............................. 66
    11.2.3 \cntmApreset: controlling continuum arcs and wedges ....... 67
  11.3 \istrootcntm and \istrootcntmA: oval continuum versions ... 69
11.4 Doing some chores: not for most users ..................... 70
  11.4.1 \cntmista: controlling outermost branches .............. 70
  11.4.2 \cntmAistb: controlling outermost branches ............... 71
  11.4.3 \xtShowEndPoints* .................................... 72
11.5 Information sets for continua of branches .................. 73
  11.5.1 Information sets for continuum triangles ............... 73
  11.5.2 \cntmAinfosett for continuum arcs ................... 73
  11.5.3 \cntmAinfosettO: oval version ....................... 74
  11.5.4 Fine-tuning \cntmAinfosettO: Not for most users .......... 75
11.6 Transition from obsolete macros \istcntmA and \istcntmarc .... 75
12 Arrows ....................................................... 77
  12.1 Using TikZ arrow options with \istgame .................. 77
    12.1.1 TikZ arrow option \rightarrow ........................ 77
  12.2 Middle arrows for a branch ................................ 78
    12.2.1 A middle arrow for each \istb ......................... 78
    12.2.2 Middle arrow tip styles ................................ 78
    12.2.3 \setxtarrowtips: controlling middle arrow tips via \rightarrow .... 79
  12.3 Middle arrows for simple trees ............................ 79
    12.3.1 \xtShowMidArrows and \xtHideMidArrows ................ 79
    12.3.2 \setxtshowmidarrows: controlling \xtShowMidArrows ......... 80
  12.4 Arrows for simple trees ................................... 81
    12.4.1 \xtShowArrows and \xtHideArrows ........................ 81
    12.4.2 \setxtshowarrows: controlling \xtShowArrows ............. 82
    12.4.3 Arrows and continua of branches: examples (Not for most users) ... 82
13 Supplementary macros ......................................... 83
  13.1 Supplementary macros to important labels .................. 83
  13.2 More supplementary macros ................................. 85
    13.2.1 \xtTimeLineH and \xtTimeLineV ......................... 85
    13.2.2 \xtCommentTo and \xtCommentFrom ........................ 87
14 Representing subgames ....................................... 88
  14.1 \xtSubgameBox (experimental) ............................ 88
    14.1.1 \xtSubgameBox ....................................... 88

iii
0 Changes and remarks

0.1 Changes in version 2.1

Version 2.1 of the istgame package introduces a starred version of the istgame environment, some new macros, and some examples added to the documentation.

**The starred version of istgame environment:** The new starred version of the istgame environment is essentially the same as the tikzpicture environment.

- The standard version of istgame environment checks the existence of TikZ scaling and the arrow option $\rightarrow$, and uses the collected information to automatically get the best results. (See Section 3.1 on page 13 for more details.)
- \begin{istgame}* may be slightly faster than \begin{istgame} as it does not collect this information.
- However, using \begin{istgame}* may require some manual work to get desired results, especially for asymmetric TikZ scaling (i.e. when \texttt{xscale} is not the same as \texttt{yscale}).
- With \begin{istgame}*:
  - \texttt{\xtxscale}, \texttt{\xtyscale}, and \texttt{\xtscale} have no effect.
  - \texttt{\xtcureslopedlabelsNS} and \texttt{\xtcureslopedlabelsEW} do nothing.
  - The oval type information sets \texttt{\xtInfosetO}, \texttt{\xtCIfnosetO}, and \texttt{\cntmAInfosetO} may be distorted.
  - \texttt{\setistgamefontsize} and \texttt{\setistgameshorten} have no effect.
  - The initial TikZ arrow style (but not $\ggstealth$) is used.

It is recommended to use the standard istgame environment, when you use the oval type information sets and sloped labels with TikZ scaling. **In all other cases, \begin{istgame}* and \begin{istgame} will give you the same results in drawing tree structures (except for the default arrow style).**

Here are some examples to show that the istgame environment can automatically achieve the best results.

```latex
% asymmetric scaling: distorted oval information sets
\begin{istgame}*[xscale=1.5,yscale=.9] \% starred
\setistgrovdirection'{south east}
\istroot(0) \istb \istb* \endist
\istroot(1)(0-1){2} \istb* \istb* \endist
\xtInfosetO[fill=red!20,ellipse](0)(0-2){1}
\xtInfosetO(1)(1)
\xtInfosetO[fill=blue!40,opacity=.5](1-1)(1-2){3}(1.5em)
\end{istgame}

% Oval information sets do not depend on scaling
\begin{istgame}[xscale=1.5,yscale=.9] \% starred
\setistgrovdirection'{south east}
\istroot(0) \istb \istb* \endist
\istroot(1)(0-1){2} \istb* \istb* \endist
\xtInfosetO[fill=red!20,ellipse](0)(0-2){1}
\xtInfosetO(1)(1)
\xtInfosetO[fill=blue!40,opacity=.5](1-1)(1-2){3}(1.5em)
\end{istgame}
```
The \texttt{tikz} library does not seem to deal with the sloped labels correctly in the case of asymmetric scaling. (See page 23 for more details.)

\begin{verbatim}
\begin{istgame}*[xscale=2,font=\footnotesize] % starred
\xtcureslopedlabelsNS % does nothing
\istroot(0)
 \istb[dashed,thick]{Left}[above,sloped]
 \istb[->]{Center}[above,sloped]
 \istb[draw=blue,thick]{Right}[above,sloped]
\endist
\end{istgame}
\end{verbatim}

The macro \texttt{\xtcureslopedlabelsNS} solves this problem only when you use the standard version of \texttt{istgame}.

\begin{verbatim}
\begin{istgame}[xscale=2,font=\footnotesize]
\xtcureslopedlabelsNS
\istroot(0)
 \istb[dashed,thick]{Left}[above,sloped]
 \istb[->]{Center}[above,sloped]
 \istb[draw=blue,thick]{Right}[above,sloped]
\endist
\end{istgame}
\end{verbatim}

New macros to set the default node sizes: In order to change the size of basic nodes, new macros \texttt{\setistsolidnodesize}, \texttt{\setisthollownodesize}, \texttt{\setistellipsenodesize}, and \texttt{\setistrectanglenodesize} are introduced. You can use these macros outside of the \texttt{istgame} environment (or in the preamble) to change the default node size globally.

In the following example you can see the effect of \texttt{\setistsolidnodesize{.5\pgflinewidth}}. The default size of a solid node is 2.4pt.

\begin{verbatim}
\setistsolidnodesize
\begin{istgame}
\istroot(0)(0,0) \istb\istb\istb \endist
\istroot(c)(0-3) \istb\istb \endist
\setistsolidnodesize{.5\pgflinewidth} % solid node size changed
\istroot(0)(6,0) \istb\istb\istb \endist
\istroot(c)(0-3) \istb\istb \endist
\end{istgame}
\end{verbatim}

\textbf{Some more}: The update to version 2.1 includes a few bug fixes and minor changes. (See the version history on page 129.)

The \texttt{TikZ} libraries \texttt{arrows.meta} and \texttt{bending} are added to the the list of preloaded \texttt{tikz} libraries. You can see some examples of arrow tips from \texttt{arrows.meta} in Section 12.2.3 on page 79.
0.2 Changes in version 2.0

Some macros have been changed and removed in version 2.0. Those who have used these changed and removed macros may want to find and replace the followings:

<table>
<thead>
<tr>
<th>ver. 1.0</th>
<th>ver. 2.0 or later</th>
</tr>
</thead>
<tbody>
<tr>
<td>\istb.</td>
<td>\istbt</td>
</tr>
<tr>
<td>\xtInfoset'</td>
<td>\xtInfoset</td>
</tr>
<tr>
<td>\xtInfosetO'</td>
<td>\xtInfosetO</td>
</tr>
<tr>
<td>\setistgrowkey</td>
<td>\setxtgrowkey</td>
</tr>
</tbody>
</table>

Changed and removed macros

- The macro name \istb. (terminal version) has been changed to \istbt (terminal version).
  - This is the opportunity cost of having a new macro \istB (dual action label version).
- The two (unsatisfactory) macros \xtInfoset' and \xtInfosetO' have been removed.
  - The macro \xtInfoset0 is completely redesigned, so that we do not need the macro \xtInfosetO' any more.
  - No reasons could be found to keep (even for the backward compatibility) the swap versions \xtInfoset' and \xtInfosetO', except for the inconvenience to do ‘find and replace.’
- \setistgrowkey is renamed to \setxtgrowkey for consistency in macro naming.

Redesigned macros and the environment

- \xtInfoset0 is completely redefined to improve its function:
  - Now a sloped information set is possible.
  - It connects two nodes like \xtInfoset0(coor1)(coor2), but if the two coordinates are identical it represents a singleton information set by a circle by default.
  - This change does not seem to cause much harm, but be aware that the swap version \xtInfosetO' has been removed and replaced by \xtInfoset0.
  - Be aware also that the way to change the height (1em by new default) of an information set has been changed, though you might not see much difference if you have only used the default information sets.
  - With new macros \xtCInfoset and \xtCInfoset0, a curved (even skewed curved) information set is now possible.
- istgame environment: (internal change)
  - Now the each value of the option of xscale and yscale, if exists, is extracted and saved at \xtxscale and \xtyscale, respectively. The value of scale is also saved at \xtscale only when it is used without xscale nor yscale. These values are internally used to get the best outputs of trees in many ways.
  - If the Ti\texttt{k}Z arrow option \texttt{-}> exists in the option list of an istgame environment, you can globally control the arrow-end-shorten value (by default, shorten \texttt{>=}0pt) by using a new macro \setistgameshorten. This is to get a better result of branches with arrows.
- Some changes that you might not notice have been made, including:
  - The core macros \istroot, \istb, and \endist have literally been redefined for some purposes, but this makes no difference to users.
  - The options thin and solid have been added to the definitions of basic node styles.
  - Some default values have been slightly changed.
0.3 What’s new in version 2.0

Some new functions

- **input mode changer** (math or text) for important labels: owners, action labels, and payoffs
  - \setistmathTF as an input mode changer for the important labels
  - \setistmathTF* having additional functions as a *text font style changer*

- **curved (even skewed) information sets**
  - \xtCInfoset for curved information sets
  - \xtCInfosetO for curved bubble type information sets

- **enhanced continuum of branches** (making \istcntm and \istcntmarc obsolete)
  - \istrootcntm for a continuum triangle
  - \istrootcntmA for a continuum arc (with \istbA)
  - \cntmAInfoset and \cntmAInfosetO for information sets for a continuum of branches

- **arrows and middle arrows** on branches
  - controllable arrow option -> with \setxtarrowtips and middle arrow tip styles
  - \xtShowMidArrows and \xtShowArrows

- and some more
  - \istB for dual action labels
  - \xtTimeLineH, \xtTimeLineV, \xtCommentTo, \xtCommentFrom, etc.

List of new macros

- \istbt(*): terminal version of \istb(*) (replacement of the removed macro \istb.(*))
- \istB(*): dual action label version of \istb(*)
- \istBt(*): terminal version of \istB(*)
- \istbA(*): alternative version of \istb(*) (intended to work with a continuum arc)
  - \istbAt(*): terminal version of \istbA(*)
- \setistmathTF: input mode changer (math or text) for owners, action labels, and payoffs
  - \setistmathTF*: input mode and font style changer for owners, action labels, and payoffs
- \xtCInfoset: (curved version) curved information set
- \xtCInfosetO: (curved oval version) curved oval type information set
- \xtCInfosetOTurnX: turns X circles of \xtCInfosetO (to use it just in case)
- \xtInfoset*: prints owners according to the input mode as set by \setistmathTF(*)
  - \xtInfoset0*: prints owners according to the input mode as set by \setistmathTF(*)
  - \xtInfoset*: prints owners according to the input mode as set by \setistmathTF(*)
  - \xtInfoset0*: prints owners according to the input mode as set by \setistmathTF(*)
  - \xtOwner*: prints owners according to the input mode as set by \setistmathTF(*)
  - \xtActionLabel*: prints action labels according to the input mode set by \setistmathTF(*)
  - \xtPayoff*: prints payoffs according to the input mode as set by \setistmathTF(*)
  - \xtInfosetOwner*: prints owners according to the input mode set by \setistmathTF(*)
  - \setxtinfosetstyle: changes line the style (line style, fill, etc.) for all information sets
  - \setxtinfosetlayer: changes the layer of information sets
  - \setxtsubgamelayer: changes the layer of \xtSubgameBox and \xtSubgameOval
  - \setistgameshorten: value of the key shorten > in istgame environment option list
- \cntmdistance: analogous to \xtdistance, for a continuum of branches
  - \cntmdistance*: incorporates \cntmdistance with \xtdistance
  - \istrootcntm: \istroot + cntm, printing a continuum of branches
  - \istrootcntm*: swap version of \istrootcntm
  - \istrootcntm*: (oval version) \istroot + cntm, printing a continuum triangle
  - \istrootvecntm*: swap version of \istrootcntm
  - \cntmpreset: controls a continuum of branches (line style, color, size, fill color)
  - \cntmpreset*: controls a simple triangle continuum of branches with no background color
• \cntmistb: similar to \istb for the outermost branches of a continuum triangle
• \cntmistb*: draws solid nodes at the ends of the continuum outermost branches
• \istrootcntmA: \istroot + \cntmA, printing a continuum arc
• \istrootcntmA*: swap version of \istrootcntmA
• \istrootocntmA: (oval version) \istrooto + \cntmA, printing a continuum arc
• \istrootocntmA*: swap version of \istrootocntmA
• \cntmapreset: controls the features of a continuum arc
• \cntmAlayerpreset: sets the layer of a continuum wedge (with fill color)
• \cntmAistb: similar to \istb for the outermost branches of a continuum arc
• \cntmAistb*: draws solid nodes at the ends of the continuum arc outermost branches
• \cntmAInfoset: prints an information set for a continuum arc
• \cntmAInfosetO: oval version of \cntmAInfoset
• \xtShowEndPoints*: shows additionally the outermost endpoints of a continuum of branches
• \xtHideEndPoints*: turns off only the endpoints of a continuum of branches
• \cntmAexpostShowEndPoints: shows the two endpoints of a continuum arc
• \setxtarrowtips: works through \textendash{} to control the features of middle arrow tips
• \xtShowMidArrows: shows arrows in the middle of branches
• \xtHideMidArrows: hides middle arrows drawn by \xtShowMidArrows
• \setxtshowmidarrows: controls the middle arrows on branches in a simple tree
• \xtShowArrows: shows arrows at the ends of branches with endpoints
• \xtHideArrows: hides arrows drawn by \xtShowArrows
• \xtHideArrows*: hides arrows but with endpoints remained
• \setxtshowarrows: controls the features of the arrows shown by \xtShowArrows
• \xtTimeLineH: a horizontal time-line
• \xtTimeLineH*: a horizontal time-line with a label at the other end
• \xtTimeLineV: a vertical time-line
• \xtTimeLineV*: a vertical time-line with a label at the other end
• \xtCommentTo: to leave a comment to a node from a relative coordinate
• \xtCommentFrom: to leave a comment from an absolute coordinate to a node
• \xtcureslopedlabelsNS: cures the Ti\LaTeX{} issue of sloped labels with asymmetric scales
• \xtcureslopedlabelsEW: same, for a tree growing eastwards or westwards

The last two macros are for only temporary use and could be removed at any time.

List of new arrow styles
• \textendash{}: (controllable) middle arrow tip, taking one optional argument
• \textendash\textendash\textendash{}: double middle arrow tip
• \textendash\textendash\textendash\textendash{}: triple middle arrow tip
• \textendash{}: circle middle arrow tip
• \textendash\textendash{}: cross middle arrow tip

0.4 How to read this document
As a Ti\LaTeX{} user, if this is your first read of this manual, all you need to read is Section 1 entitled “Getting started.” That’s only four pages long. (You can find a little secret at the first part of Section 1.3 entitled “Complete examples for desperate users.”) Every example in this document is provided with its complete codes you can copy to use. Now you can go to Section 1 to get started!

If you are not urgent, you can continue to read sections on core macros up to Section 5. That’s about twenty pages total. Still more time? Then read Section 6 (entitled “Important labels: players, action labels, and payoffs”) and Section 7 (entitled “Input mode and text font style changer”). That’s about thirty pages all total.

Throughout the manual, just disregard the parts with marks including “fine-tuning” or “not for most users,” if you are not an experienced user of this package. Just reading first part of each section will suffice most users to draw game trees in almost all cases, hopefully.

If you you are an experienced user of the istgame package, enjoy every detail of the package.
0.5 Remarks

Rules of thumb for the usage of delimiters  Though the rules are not strictly observed, it might be useful to go over the rules of thumb for the usage of delimiters.

- { }: contents, texts, important dimensions (mandatory or optional)
- [ ]: usual options, positions or directions, node types, fill color
- ( ): coordinate related arguments, dimension as the last optional argument, special purposes
- < >: angles (or directions), special purposes (mostly used right before { })
- + .. +: only for the local change of level and sibling distances with \istroot and its friends
- ! !: midpoint factor only for curved information sets

As for the order of delimiters, the (somewhat loose) rules are as follows:

- Generally, all macros are designed to avoid leaving an empty argument like [] [blue], as possible as they can do. (This is input minimalism for lazy users including myself!)
- < > is used just before { }, except for \istb and its friends.
- ( ) as the last optional argument is for dimensions, like (1em) or (3pt).
- The order mostly looks like <=}{<} or <=}{<}, especially after a mandatory argument.

Optional versions of macros  Some macros have a starred(*) version or a swap(') version, for example, \istb* or \istroot'.

- * version: just another version with different functions (in some cases, it’s quite different)
- ' version: clockwise arrangement of branches or its related version

Global macros  Some macros have global effects, so you can use them outside of the istgame environment or even in the preamble of your document, but with very great caution.

The following macros, all prefixed by \setist..., can be used outside of the istgame environment, to change the default values of the options:

- \setistgamefontsize{<text size>}  (default: \normalsize)
- \setistgameshorten{<arrow end shorten dim>}  (default: 0pt)
  - This works only when -> exists in the option list of the istgame environment.

Though it is not recommended, you can use all the macros prefixed by \setist... inside or outside of the istgame environment to change the default values.

- \setistmathTF, \setistmathTF*  (initially: 011)
- \setistdefaultnodeinnersep
- \setistdefaultnodeoutersep
- \setistdefaultnodedefdrawcolor  (default: black)
- \setistdefaultnodefillcolor  (default: white)
- \setist<...>NodeStyle
- \setistgrowdirection, \setistgrowdirection'  (default: south):
  - You may not want to use (but still can use) this outside of an istgame environment.

So, for example, you can do, in the preamble of your document, like:

\usepackage{istgame}
\setistgamefontsize{\normalsize}
\setistdefaultnodedefdrawcolor{black}
\setistsolidnodesize{2.4pt} \% (new in version 2.1)
\setistellipsenodesize{2.8pt} \% (new in version 2.1)
\setistgameshorten{.3pt}
\setistmathTF011
The node styles such as \texttt{ellipse node} and \texttt{rectangle node} has white as the default background color. This means that if the paper color of your document is not white, say, blue!16, you might not satisfy the results because the nodes have white background. In this case you can resolve the conflict by adding the following two lines at the very first of your document:

\begin{verbatim}
\papercolor{blue!16}
\setistdefaultnodefillcolor{blue!16}
\end{verbatim}

**Known problem with the \texttt{tikz-qtree} package** It seems that \texttt{tikz-qtree} changes node anchors. So, with the \texttt{tikz-qtree} package uploaded, you will get unexpected results when you draw a game tree by using the native \texttt{tree} library in TikZ. Since the \texttt{istgame} package is based on the \texttt{tree} library, it is also affected by \texttt{tikz-qtree}, resulting in unexpected outputs.

The best way to resolve this problem is that you DO NOT LOAD \texttt{tikz-qtree} when you draw game trees with TikZ. If, for some reason, you need to load \texttt{tikz-qtree} when you draw a game tree by using the \texttt{istgame} package, a temporary solution to resolve the conflict is to add the TikZ option \texttt{edge from parent path} in the option list of \texttt{istgame} environment as follows:

\begin{verbatim}
% tikz-qtree conflict resolution (only with \usepackage{tikz-qtree})
[
  edge from parent path={("\tikzparentnode") -- ("\tikzchildnode")}
]
\end{verbatim}

0.6 Previous changes (up to version 1.0)

A considerable number of macro names have been changed in the version 0.8 (Jan. 17, 2017) of this package.\footnote{The \texttt{istgame} package of the version older than ver. 1.0 had been distributed via the KTUG (Korean TeX Users Group) Private Repository.} The following old macro names in any previously written documents using codes in \texttt{istgame} ver. 0.7 or before, should be replaced by the new names, accordingly.

Also, \texttt{\textbackslash istroot*}, \texttt{\textbackslash istcntm*}, and \texttt{\textbackslash xtInfoset*} must be replaced by \texttt{\textbackslash istrooto}, \texttt{\textbackslash istcntmarc}, and \texttt{\textbackslash xtInfosetO}, respectively, in the version 1.0 or later.

<table>
<thead>
<tr>
<th>\textbackslash xdistance</th>
<th>\textbackslash xtdistance</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbackslash xDot</td>
<td>\textbackslash xNode</td>
</tr>
<tr>
<td>\textbackslash xInfoset</td>
<td>\textbackslash xtInfoset</td>
</tr>
<tr>
<td>\textbackslash xInfoset*</td>
<td>\textbackslash xtInfoset*</td>
</tr>
<tr>
<td>\textbackslash xInfosetOwner</td>
<td>\textbackslash xtInfosetOwner</td>
</tr>
<tr>
<td>\textbackslash xActionLabel</td>
<td>\textbackslash xtActionLabel</td>
</tr>
<tr>
<td>\textbackslash xPayoff</td>
<td>\textbackslash xtPayoff</td>
</tr>
<tr>
<td>\textbackslash ShowTerminalNodes</td>
<td>\textbackslash xtShowTerminalNodes</td>
</tr>
<tr>
<td>\textbackslash HideTerminalNodes</td>
<td>\textbackslash xtHideTerminalNodes</td>
</tr>
<tr>
<td>\textbackslash levdist</td>
<td>\textbackslash xtlevdist</td>
</tr>
<tr>
<td>\textbackslash sibdist</td>
<td>\textbackslash xtSibdist</td>
</tr>
<tr>
<td>\textbackslash \setistactionlabelshift</td>
<td>\textbackslash xtALPush</td>
</tr>
<tr>
<td>\textbackslash \setistactionlabelposition</td>
<td>\textbackslash xtALShift</td>
</tr>
<tr>
<td>\textbackslash \istroot*</td>
<td>\textbackslash istrooto</td>
</tr>
<tr>
<td>\textbackslash \istcntm*</td>
<td>\textbackslash istcntmarc</td>
</tr>
</tbody>
</table>
1 Getting started

The package istgame provides macros built on TikZ to draw game trees. The core macros provided with this package are \texttt{\textbackslash istroot}, \texttt{\textbackslash istb}, and \texttt{\textbackslash endist}. \texttt{\textbackslash istroot} pins down the root of a tree or a subtree, \texttt{\textbackslash istb} represents a branch, and \texttt{\textbackslash endist} indicates the end of drawing a simple tree. Without \texttt{\textbackslash endist}, the tree is NOT actually drawn, with no error messages produced. A tree drawn by the sequence of \texttt{\textbackslash istroot-\textbackslash istb-\textbackslash endist} is a simple tree. You can draw a whole game tree by repeatedly connecting these simple tree structures.

Here, the prefix ‘ist’ stands for ‘it’s a simple tree.’ You can also read it as ‘insung’s simple tree’ if you would like.

The package istgame depends on the packages tikz, xparse, and expl3.

To use the istgame package you must load the package in the preamble of your document:

\begin{lstlisting}[language=TeX]
\usepackage{istgame}
\end{lstlisting}

The package preloads the following TikZ libraries:

\begin{itemize}
  \item trees,\texttt{calc},\texttt{arrows},\texttt{shapes},\texttt{positioning},\texttt{backgrounds},\texttt{fit},\texttt{decorations.markings},\texttt{arrows.meta},\texttt{bending},
\end{itemize}

and also preloads \texttt{patterns} and \texttt{intersections} for additional use.

1.1 Getting-started example: a simple tree

Let us get started with a simple self-explanatory example:

\begin{lstlisting}[language=TeX]
\begin{istgame}
\istroot(0)(0,0) % names the root as (0) at (0,0)
  \istb % endpoint will be (0-1), automatically
  \istb % endpoint will be (0-2), automatically
  \istb % endpoint will be (0-3), automatically
\endist % end of simple (parent-child) structure
\end{istgame}
\end{lstlisting}

The resulting tree has the \textit{height} of 15mm and the \textit{distance between two neighbor endpoints} (not shown) is also 15mm by default. In TikZ, the height is called the \textit{level distance} and the distance between two neighbor endpoints is called the \textit{sibling distance}.

If the second parenthesis argument of \texttt{\textbackslash istroot} is omitted, it is regarded as (0,0) by default, otherwise it is necessary to specify the coordinate from which a simple tree starts.

1.2 Connecting simple tree structures

Basically, in order to draw a whole game tree, we just repeat the simple \texttt{\textbackslash istroot-\textbackslash istb-\textbackslash endist} structure.

\begin{lstlisting}[language=TeX]
\begin{istgame}
\istroot(0) % names the root (0) at (0,0)
  \istb % endpoint will be (0-1), automatically
  \istb % endpoint will be (0-2)
  \istb % endpoint will be (0-3)
\endist % end of simple (parent-child) structure
\istroot(c)(0-3) % names the subroot (c) at (0-3)
  \istb % endpoint will be (c-1)
  \istb % endpoint will be (c-2)
\endist
\end{istgame}
\end{lstlisting}
In the above example, the simple subtree is rooted at (0–3), names the subroot (c), and has two branches whose endpoints are automatically named (c–1) and (c–2), respectively.

Note that the user-defined names of the (sub)roots and the names of endpoints are arranged counterclockwise (from left to right) by TikZ at the endpoints of branches, which can be used as coordinates in the usual TikZ way.

1.3 Complete examples for desperate users

Basically, \texttt{\textbackslash istroot} designates a decision node and its owner (or a player), \texttt{\textbackslash istb} prints a branch coming out from the decision node with action labels and payoffs, and \texttt{\textbackslash endist} actually draws the tree structures. (One secret is that I almost always use only the basic features of the istgame package, discussed in this subsection.)

1.3.1 How to put a decision node and its owner

\[
\texttt{\textbackslash istroot}(<\text{decision node name}>)(<\text{root location}>)<\text{owner position}>{<\text{owner}>}
\]

The only mandatory argument of \texttt{\textbackslash istroot} is \texttt{(decision node name)} and all others are optional. If the \texttt{(location)} where the root or a decision node is placed is omitted, it is regarded as \texttt{(0,0)} by default. The position of an owner (or a player) is \texttt{<above>} (or equivalently, \texttt{<90>}) degree by default. The owner of a node is printed in text mode by default.

% Example: first try
\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot(0)(0,0){Child}
 \istb \istb \endist
\istroot(1)(0–2)<30>{Parent}
 \istb \istb \endist
\end{istgame}

\begin{tikzpicture}
  \node (root) at (0,0) {Child};
  \node (parent) at (30:3cm) {Parent};
  \draw (root) -- (parent);
\end{tikzpicture}

In fact, \texttt{\textbackslash istroot} and its variants have much more functions than these. Later, you can look into Section 4 on page 16 for more details.

1.3.2 How to print branches with action labels and payoffs

\[
\texttt{\textbackslash istb}(<\text{action label}>)[<\text{action label pos}>]{<\text{payoffs}>}[<\text{payoff pos}>]
\]

With the macro \texttt{\textbackslash istb}, you can draw a branch and put an action label and payoffs as optional arguments.

% Example: first try
\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot(0){Child}
 \istb{Good}[above left]{(0,2)}
 \istb{Bad}[above right]
 \endist
\istroot(1)(0–2)<30>{Parent}
 \istb{Forgive}[all]{(1,1)}
 \istb{Punish}[ar]{(-1,-1)}
 \endist
\end{istgame}

% Example:
\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot(0){Child}
 \istb{Good}[above left]{(0,2)}
 \istb{Bad}[above right]
 \endist
\istroot(1)(0–2)<30>{Parent}
 \istb{Forgive}[all]{(1,1)}
 \istb{Punish}[ar]{(-1,-1)}
 \endist
\end{istgame}

\begin{tikzpicture}
  \node (child) at (0,0) {Child};
  \node (parent) at (30:3cm) {Parent};
  \draw (child) -- (parent);
  \node (good) at (90:1.5cm) {Good};
  \node (bad) at (210:1.5cm) {Bad};
  \node (forgive) at (-45:1.5cm) {Forgive};
  \node (punish) at (135:1.5cm) {Punish};
  \draw (child) -- (good) -- (bad);
  \draw (child) -- (forgive) -- (parent);
  \draw (child) -- (punish) -- (parent);
\end{tikzpicture}
The positions of an action label and payoffs are specified as options right after each of the two. When you omit the position of payoffs, the package prints them naturally, but still you can change the location using an option like \texttt{[left]} or \texttt{[above left]}. For the positions of action labels and payoffs, you can use an abbreviation \texttt{[al]} for \texttt{[above left]} and similarly \texttt{[ar]}, \texttt{[bl]}, and \texttt{[br]}. The abbreviations \texttt{[a]}, \texttt{[b]}, \texttt{[l]}, and \texttt{[r]} are also available.

Both of the action labels and payoffs are printed in \textit{math mode} by default.  (You can change the input mode for owners, action labels, and payoffs using a very useful macro \texttt{\setistmathTF}, documented in Section \ref{sec:math} on page \pageref{sec:math}.)

In fact, \texttt{\istb} and its variants have much more functions than these.  For more details, see Section \ref{sec:full}, on page \pageref{sec:full}.  If this is your first read of this manual, however, you don’t need to bother about all the details at the moment.

Following three sections are about an information set, a continuum of branches, and changing the direction of tree growing.  If you do not need to use them now, the core macros \texttt{\istroot}, \texttt{\istb}, and \texttt{\endist} are all you need to know about drawing game trees of any size, small or big. Just connect simple trees to complete a whole tree.

### 1.3.3 How to put information sets

The macro \texttt{\xtInfoset} connects two nodes with a densely dotted line, by default, representing an information set.

\begin{verbatim}
\xtInfoset(<from coor>)(<end coor>){<owner>}[<owner pos>]
\end{verbatim}

The two node coordinates are \textit{mandatory}.  You can put the owner of an information set as an optional argument.  An owner is printed above (by default) the line in \textit{text mode}, which can be changed.  For more details about \texttt{\xtInfoset}, see Section \ref{sec:info} on page \pageref{sec:info}.

In the example below, the macro \texttt{\xtInfoset} is used to show an information set.

\begin{verbatim}
% Example: information set
\begin{istgame}
\setistgrowdirection'\{east\}
\xtdistance{15mm}{30mm}
\istroot(0){Alice}
  \istb(A){al}{(2,2)}
  \istb(D){ar}
\endist
\istroot(1)(0-2)<above right>{Alice}
  \istb(L){al}
  \istb(R){ar}
\endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)
  \istb(\ell){al}{(4,2)}
  \istb(r){ar}{(1,1)}
\endist
\istroot(3)(1-2)
  \istb(\ell){al}{(3,2)}
  \istb(r){ar}{(0,3)}
\endist
\xtInfoset(2)(3){Elaine}
\end{istgame}
\end{verbatim}

The package also provides the macro \texttt{\xtInfosetO} to draw a bubble type information set.  Just replace \texttt{\xtInfoset} with \texttt{\xtInfosetO} (see Section \ref{sec:info} on page \pageref{sec:info}).

You can also draw a curved information by using \texttt{\xtCInfoset} (Section \ref{sec:curve}) and even a curved bubble type by using \texttt{\xtCInfosetO} (Section \ref{sec:curve}).  You can try now.
1.3.4 How to put a continuum of branches

Just use \istrootcntm (\istroot + cntm), instead of \istroot, to draw a continuum of branches.

\istrootcntm(<decision node name>)(<root location>){<owner position>{<owner>}}

The macro \istrootcntm works just like \istroot, but it prints a background triangle in black!25, by default, representing a continuum of branches.

% Example: continuum of branches
\begin{istgame}[font=\scriptsize]
  \istrootcntm(0){1}
  \istb{x}[r] \istbm \endist
  \xtdistance{10mm}{20mm}
  \istroot(1)(0-1)<120>{2}
  \istb{Y}[al]{x,1-x} \istb{N}[ar]{0,0} \endist
\end{istgame}

Here, \istbm represents a missing (or an invisible) branch (see page 22).
You can also change the color and size of the triangle representing a continuum of branches. For more details on \istrootcntm see Section 11.1, on page 60.

The package also provides the macro \istrootcntmA to draw an arc to represent a continuum of branches. You can just replace \istrootcntm by \istrootcntmA to do that, but let us not try this now. (If you really want to try this now, you should change the first \istb to \istbA and then take out \istbm in the above example. It is said not to try this now.) For more details on \istrootcntmA see Section 11.2, on page 65.

You can use every options and macros you can use for the \tikzpicture environment with the \istgame environment. In the above example, font=\scriptsize is used as an option.

1.3.5 How to change the growing direction of a tree

With \setistgrowdirection or \setistgrowdirection', you can easily change the direction (south by default) to which a game tree grows, as shown in the example below.

% Example: \setistgrowdirection(')
\begin{istgame}[scale=1.2]
  \setistgrowdirection'{east}
  \istroot(0)<180>{1}
    \istb(a)[a] \\
    \istb(b)[a] \\
    \istb(c)[b] \\
  \endist
  \xtdistance{10mm}{20mm}
  \istroot(1)(0-2)<135>{2}
    \istb{Y}[al]{x,1-x} \\
    \istb{N}[ar]{0,0} \\
  \endist
\end{istgame}

All you need to do is just to specify the direction you want, like \setistgrowdirection(west), \setistgrowdirection'{east}, or \setistgrowdirection'{-45}. The prime version ('') is just to arrange branches clockwise (by default counterclockwise). In the above example, if you use \setistgrowdirection without the prime, the branches will be arranged counterclockwise, like a, b, and c from bottom to top. When changing the direction of a tree, you may want to relocate the owner and action labels. For more details, see Sections 9 (on page 45) and 6.3.2 (on page 37).

Now you are ready to draw any standard game trees such as all game trees in Osborne’s book.
2 Important distances: \xtdistance

The length and direction of branches in a simple tree can be controlled by the macro \xtdistance. Here, the prefix ‘xt’ stands for extensive tree.

The macro \xtdistance sets or resets the level distance and the sibling distance, respectively. Note also that internally, for example, \xtdistance{20mm}{30mm} assigns 20mm to \xtlevdist and 30mm to \xtsibdist, which renew the default distances. It is effective until you change the distances by using another \xtdistance.

You can use \xtdistance at any time you want to change the length and the directions of branches. Since we are dealing with simple parent-child tree structures, \texttt{<level depth>} is 1 by default. (The level depth number other than 1 is not expected to be used.)

Remark:

- Since \xtlevdist and \xtsibdist are assigned values by \xtdistance, you can use these values to do some calculation like, for example, 1.2*\xtlevdist or 1.5*\xtsibdist.

- The starred version \texttt{\cntmdistance*} is provided to deal with a continuum of branches and \xtdistance together, which is not documented at the moment. See Section 11.1.3, on page 63, for more details about \cntmdistance*.

In fact, the core macros are much more powerful. \istroot controls the direction to which a simple parent-child tree grows, node styles, the node owner and its position, the height and sibling distance of a current simple tree, etc. \istb specifies the growing direction of an individual branch, branch line styles, branch color, action labels, and payoffs and their position. Below we will see in more details on how the core macros and others work.
3 The istgame environment and node styles

3.1 The istgame environment

The package provides the istgame environment, which is basically the sum of the \texttt{tikzpicture} environment plus some additional functions and different initial values. So it accepts all the options and macros that can be used for the \texttt{tikzpicture} environment. (Note that most of the macros provided by the package work also in the \texttt{tikzpicture} environment, but some works only in the istgame environment.)

\begin{verbatim}
\def\istgame@default@fontsize{\normalsize}
\begin{istgame}[<Tikz options>]
  <istgame contents>
  <tikzpicture contents>
\end{istgame}
\end{verbatim}

The default font size is set as \texttt{font=\normalsize}. You can globally change the default font size by using \texttt{\setistgamefontsize}, like \texttt{\setistgamefontsize{\scriptsize}}. Since the environment \texttt{istgame} is basically the same as \texttt{tikzpicture}, you can also locally change the font size by using the \texttt{font} option key, like \texttt{\begin{istgame}[font=\scriptsize]...\end{istgame}}.

Remark: (Not for most users) What the \texttt{istgame} environment \textit{internally} does includes:

- The \texttt{istgame} environment internally checks and extracts the optional values of \texttt{xscale} and \texttt{yscale} and, if exist, saves the values (1.0 by default) at \texttt{\xtxscale} and \texttt{\xtyscale}, respectively. And if the optional value of \texttt{scale} exists, it is saved at \texttt{\xtscale} only when neither \texttt{xscale} nor \texttt{yscale} exists. You can use these values to calculate something you want, like 5*1/\texttt{\xtscale}. The extracted values are internally used to get the best results of the shapes of bubble type information sets.

- The \texttt{istgame} environment also checks if the arrow option \texttt{[->]} exists in the option list of the environment. If it exists the \texttt{istgame} adds \texttt{shorten =\dimen} (by default 0pt) to the list as the first option together with \texttt{[->]}. You can change the arrow shortening default value, like \texttt{\setistgameshorten{1.3pt}}. (Though this is \textit{not for most users}, you can see some more details, in Section 12.1.2.)

\textbf{Starred version} \texttt{\begin{istgame}*}: The \textit{starred version} of \texttt{istgame} environment is essentially the same as the \texttt{tikzpicture} environment. It means that if you start with \texttt{\begin{istgame}*}, the environment does not check what is said in the description above. (See the discussion on page 1 for more details on what happens in this case.)

3.2 Node styles

3.2.1 Basic node styles

The \texttt{tikzstyle}'s of the six basic node styles are predefined.

- \texttt{plain node}: draws nothing \hspace{1cm} (default: inner sep=1pt)
- \texttt{null node}: (very small node) \hspace{1cm} (default: minimum size=0.2pt)
- \texttt{solid node}: (default node style) \hspace{1cm} (default: minimum size=2.4pt)
- \texttt{hollow node}: \hspace{1cm} (default: minimum size=2.8pt)
- \texttt{rectangle node}: \hspace{1cm} (defaults: inner sep=2pt, minimum size=4pt)
- \texttt{ellipse node}: \hspace{1cm} (defaults: inner sep=1.5pt, minimum size=4.8pt)
In some special cases, you may want to change some node styles, including the minimum size. This can be done by \texttt{\setist\ldots NodeStyle}, all of whose arguments are optional.

**Syntax:**

\begin{Verbatim}
\setistPlainNodeStyle{<inner sep dim>}{<outer sep dim>}
\setistNullNodeStyle[<draw color>]{<min-size dim>}{<bg color>}{<opacity>}
\setistSolidNodeStyle[<draw color>]{<min-size dim>}{<bg color>}{<opacity>}
\setistHollowNodeStyle[<draw color>]{<min-size dim>}{<bg color>}{<opacity>}
\setistRectangleNodeStyle[<draw color>]{<min-size dim>}{<bg color>}{<opacity>}
\setistEllipseNodeStyle[<draw color>]{<min-size dim>}{<bg color>}{<opacity>}
\end{Verbatim}

% Examples:
\begin{Verbatim}
\begin{istgame}
\setistSolidNodeStyle{blue}{10pt}
\istroot (0) [solid node] \endist \end{istgame}
\begin{istgame}
\setistHollowNodeStyle{blue}{10pt}[yellow]
\istroot (0) [hollow node] \endist \end{istgame}
\begin{istgame}
\setistRectangleNodeStyle{10pt}[red][.5]
\istroot (0) [rectangle node] \endist \end{istgame}
\begin{istgame}
\setistEllipseNodeStyle{blue}{10pt}[green]
\istroot (0) [ellipse node] \endist \end{istgame}
\begin{istgame}
\setistNullNodeStyle{blue!20}{10pt}
\istroot (0) [null node] \endist \end{istgame}
\end{Verbatim}

These basic node styles have their aliases, for convenience, for those who are familiar with game theoretic terminology.

% aliases for game theorists
\begin{Verbatim}
\tikzset{decision node/.style=solid node}  \% decision nodes
\tikzset{terminal node/.style=solid node}  \% terminal nodes
\tikzset{initial node/.style=hollow node}
\tikzset{chance node/.style=hollow node}
\end{Verbatim}

The set of all nodes of a game tree can be partitioned into the set of decision nodes and that of terminal nodes. You can use initial node to distinguish the root (or the initial node) of a game tree from decision nodes. You can also use chance node to represent a chance node of a game tree. Additional convenient node aliases are also provided: box node, square node, and oval node.

% some more aliases
\begin{Verbatim}
\tikzset{box node/.style=rectangle node}
\tikzset{square node/.style=rectangle node}
\tikzset{oval node/.style=ellipse node}
\end{Verbatim}

For aliases, you can also change the node styles, like \texttt{\setistDecisionNodeStyle[blue]{3pt}} or \texttt{\setistBoxNodeStyle{3pt}[green][.5]}.

**3.2.2 Your own node styles:** \texttt{\setistNewNodeStyle}

You can create your own node style by \texttt{\setistNewNodeStyle}.

% \setistNewNodeStyle
% syntax:
% \setistNewNodeStyle{<style name>}[<opt>]{<minimum size>}
% \% defaults:
% {<m>}[-,circle,draw=black,fill=white,inner sep=1pt]{6mm}
The first, mandatory denoted by \{<m>\}, argument is \{style name\} to be used. The second, optional, argument is \[<options>\] to determine the style of a new node, with defaults. The third, optional, argument should be \(<minimum size>\) only in dimension, which may be frequently used.

To define, for example, a circle (by default) node with the minimum size of 3mm filled with red, you can do like this: \setistNewNodeStyle{new node}[fill=red]{3mm}. In fact, this is an abbreviation of the following TiKZ macro:

\begin{verbatim}
\tikzset{new node/.style={
  circle , draw=black , fill=red , inner sep=1pt , minimum size=3mm } }
\end{verbatim}

Here is an example of using \setistNewNodeStyle.

\begin{istgame}[scale=1.5]
\setistNewNodeStyle{new 1}
  [regular polygon,regular polygon sides=3, shape border rotate=180]
\setistNewNodeStyle{new 2}
  [regular polygon,regular polygon sides=3]
\setistNewNodeStyle{new 3}[star]
\xtdistance{15mm}{30mm}
\istroot(0)[new 1]<center>{I}
  \istb \istb \endist
\xtdistance{15mm}{15mm}
\istrooto(1)(0-1)[new 2]{II} \% \istrooto
  \istb \istb \endist
\istrooto(2)(0-2)[new 3]<center>{III}
  \istb \istb \endist
\end{istgame}

Remark: From the above example, observe that when you use \istrooto to put the owner of a node, you need to specify <center> for the position of the node owner. However, if you use the oval version \istrooto, you don’t need that. (See Section 4 on page 16 for more details on \istroot and \istrooto.)

Remark: In TiKZ, the shape of a node is independent of the scale option in tikzpicture. If you want to make the shape scaled according to the scale option, you can use the TiKZ option transform shape, as shown below. (In this kind of case, \istrooto helps, rather than \istroot.)

\begin{verbatim}
\begin{istgame}[scale=1.5]
\setistNewNodeStyle{new 1}
  [regular polygon,regular polygon sides=3, shape border rotate=180]
\setistNewNodeStyle{new 2}
  [regular polygon,regular polygon sides=3]
\setistNewNodeStyle{new 3}[star]
\xtdistance{15mm}{30mm}
\istroot(0)
  [new 1,transform shape]<center>{I}
  \istb \istb \endist
\xtdistance{15mm}{15mm}
\istrooto(1)(0-1)[new 2]{II} \% \istrooto
  \istb \istb \endist
\istrooto(2)(0-2)
  [new 3,transform shape]<center>{III}
  \istb \istb \endist
\end{istgame}
\end{verbatim}
4 Core macro: \istroot

4.1 \istroot: basics

4.1.1 \istroot – counterclockwise: standard version

The macro \istroot defines the root of a game or a subgame at a designated location, specifies the owner of the root or the subroot, and does other functions. In game theoretic terminology, \istroot designates a decision node and its owner (or a player).

% \istroot
% syntax: 
\istroot[<grow keyval>,<tree opt>](<coor1>)(<coor2>)
\[<node style>,<node opt>][<owner opt>owner label angle]{<owner>}
+<lev-distance>..<sib-distance>+
% defaults: 
[\south](<m>)(0,0)[\texttt{decision node}][\above]{\texttt{}+15mm..15mm}+
% arguments: (coor1) is mandatory, all others are optional arguments
[\texttt{grow}] % the direction of growing (default: south)
(\texttt{coor1}) % name of the (sub)root: mandatory
(\texttt{coor2}) % the (sub)root is at (coor2) (default: (0,0))
[\texttt{node style}] % node style (default: decision node)
\texttt{angle} % position of owner name (default: above)
{\texttt{owner}} % name of the owner of the (sub)root
+level dist..sibling dist+ % <defaults: 15mm,15mm>

The only mandatory argument, denoted by <m>, of \istroot is \texttt{(coor1)}, which gives the name of the root or subroot. All the other arguments are optional. The name of the (sub)root, \texttt{(coor1)}, can be referred as a normal coordinate. \texttt{(coor2)} specifies the location where the (sub)root is placed. If \texttt{(coor2)} is omitted, it is regarded as \texttt{(0,0)} by default.

The default node style of the root is a decision node, which is just a solid node. You can change the node style to any other node style such as initial node, chance node, oval node, box node, and so on.

Here is a simple example of drawing a tree structure.

% \istroot
\begin{istgame}
\istroot(0)
\\istb
\\istb
\\istb
\endist
\istroot(a)(0-1)[\texttt{chance node}]
\\istb
\\istb
\endist
\end{istgame}

naming children: counterclockwise or clockwise

In the previous example, the game tree has the root named \texttt{(0)}, located at \texttt{(0,0)} by default, which has three branches (by three \texttt{\istb}'s). Since \texttt{TikZ} arranges branches of a tree counterclockwise, by default, the endpoints of the three branches are automatically named \texttt{(0-1)}, \texttt{(0-2)}, and \texttt{(0-3)} from left to right (when a tree grows down).
The root of the subtree is named (a), located at (0-1), and has two children. Its children are automatically named (a-1) and (a-2) counterclockwise (or from left to right if the tree grows down). See the following code example with explanatory labels to see what is going on.

```latex
% \istroot (explained with labels)
\begin{istgame}[font=\scriptsize]
\istroot (0)
\begin{istb}
\istb \istb \istb \endist
\istroot (a)(0-1)
\begin{istb}
\istb \istb \endist
% \% labels: (ignore the following lines at the moment)
\setistmathTF
\xtOwner (0) {()}
\xtOwner (a) {()}
\xtPayoff* (0-1) {()}
\xtPayoff* (0-2) {()}
\xtPayoff* (0-3) {()}
\xtPayoff* (a-1) {()}
\xtPayoff* (a-2) {()}
\draw [->,ultra thick,blue!20] (-260:1.5) arc (-260:80:1.5cm) node [above,blue!30] {counterclockwise};
\end{istgame}
```

owner (or player)

The owner of a decision node (or a player) is expressed in curly braces, like \{player 1\}, and printed in text mode. The input mode and text font style of an owner can be changed by \setistmathTF(*) (see Section 7 on page 39, for more details).

The position of the owner of a decision node is specified in angle brackets, like <90>, <above>, or <north>. To specify the position of an owner you can use \texttt{degrees}, or the compass directions such as \texttt{north}, \texttt{south}, \texttt{east}, \texttt{west}, and their valid combinations. You can also use the positional words such as \texttt{above}, \texttt{below}, \texttt{left}, \texttt{right}, and their valid combinations.

Growing direction of a simple tree

The first bracket option is mainly for the direction of a simple tree ([south] by default). Internally, \texttt{[grow keyval]} typed in as the first option of \texttt{\istroot} renews the direction of tree growing by assigning its value to \texttt{\istgrowdirection}, whose default is \texttt{south}.

Remark:

- In fact, the first option of \texttt{\istroot} controls the features of a whole simple tree (but a node style), while the second bracket option controls a node style only.
- In addition to the direction of a simple tree, you can add more options to control the whole branch styles and their labels (except a node style). For example, if you want, at any reason, to draw a simple tree with all red dashed branches with red labels growing south-eastwards, you can do like \texttt{[south east,red,dashed]}.
- Be aware that the first entry in the option list must be a \texttt{directional word} for a simple tree.

The tree growing direction can be specified by \texttt{[degrees]} or by using the compass directions such as \texttt{[north]}, \texttt{[south]}, \texttt{[east]}, \texttt{[west]}, \texttt{[north east]}, \texttt{[north west]}, \texttt{[south east]}, \texttt{[south west]}. You can also use positional words like \texttt{[left]}, \texttt{[right]}, \texttt{[down]}, and \texttt{[up]}, but you cannot use \texttt{[above]} nor \texttt{[below]}. 
**Example 1: \istroot (one simple tree)**

```latex
\begin{istgame}[font=\itshape]
\istroot[right](0)<left>{\text{player 1}}
\istb \istb \endist
\end{istgame}
```

**Remark:** One thing you should remember about this is that \istgrowdirection is internally used in the definition of \istb to control the label position for payoffs. However, for the label position in TikZ, [below] and [above] are good, but not [down] nor [up]. So DO NOT USE [down] nor [up] to specify the tree growing direction.

**Example 2: \istroot (three simple trees connected)**

```latex
\begin{istgame}
\xtdistance{20mm}{20mm}
\istroot[right](0)[oval node]<left>{\text{player 1}}
\istb \istb \endist
\istroot(a)(0-1)[right]{\text{player 2}}\{ +15mm..10mm+ \\
\istb \istb \endist
\istroot[right](b)(0-2)[box node]<135>{\text{player 3}}
\istb \istb \endist
\end{istgame}
```

**local change of distances**

The last two options of \istroot specify the level distance and the sibling distance. This local change of distances is valid only for the corresponding simple tree, while the distances changed by \xtdistance are valid within the current istgame environment unless they are changed again by \xtdistance. Do not forget, when you use decimal distances, to delimit the decimal dimensions with curly braces, like +\{15.5mm\}..\{10.5mm\}+.

### 4.1.2 \istroot’ — clockwise: swap version

The macro \istroot’ is the swap version of \istroot. \istroot’ works just like \istroot but with one exception: going clockwise instead of counterclockwise. \istroot’ arranges its branches clockwise (or from right to left if the tree grows down).

Compare the following example with that of \istroot above on page 17. The two examples have exactly the same codes as each other except for one thing: either \istroot or \istroot’.

```latex
% \istroot’ (explained with labels)
\begin{istgame}[font=\scriptsize]
\istroot’(0) \istb \istb \istb \endist
\istroot’(a)(0-1) \istb \istb \endist
% labels: (ignore the following lines at the moment)
\setistmathTP000
\xtOwner(0)\{0\}
\xtOwner(a)\{(a)\}\{1\}
\xtPayoff*(0-1)\{(0-1)\}\{x\}
\xtPayoff*(0-2)\{(0-2)\}\{b\}
\xtPayoff*(0-3)\{(0-3)\}\{b\}
\xtPayoff*(a-1)\{(a-1)\}\{b\}
\xtPayoff*(a-2)\{(a-2)\}\{b\}
\draw [<-,.ultra thick,blue!20](-260:1.5)
arc (-260:80:1.5cm) node [above,blue!30] \{clockwise\};
\end{istgame}
```
We will look into this issue (of going counterclockwise or clockwise) in more detail in Section 9 (on page 45), where we discuss the tree growing direction.

If you draw a game tree growing south, you don’t need to worry about the swap version \texttt{\istroot}'. Just use \texttt{\istroot}.

### 4.2 \istrooto: oval version

#### 4.2.1 \istroot – counterclockwise

The macro \texttt{\istrooto} is the oval version of \texttt{\istroot}. This allows us to draw a bubble (by default, oval node) with a node owner (or a game player) in it. Except for this difference, \texttt{\istrooto} works just like \texttt{\istroot}. Since an owner is shown in a specified node with \texttt{\istrooto}, the option \texttt{<owner label angle>} of the standard version \texttt{\istroot} is ignored.

\begin{verbatim}
% \istroot
% syntax:
\istrooto[<grow keyval,tree opt>](<coor1>)(<coor2>)
[<node style>,<node opt>]{<owner>}<lev-distance>..<sib-distance>+
% default: only (coor1) is mandatory, <=, all others optional
[south](<m>)(0,0)[oval node]{}+15mm..15mm+
\end{verbatim}

The following two examples are the same as above with \texttt{\istroot} on pages 17 and 18, respectively, but now with the oval version \texttt{\istrooto}.

% Example 1: \istrooto (one simple tree)
\begin{istgame}[font=\itshape]
\istrooto[right](0)<180>{player 1}
\istb \istb \endist
\end{istgame}

% Example 2: \istrooto (three simple trees connected)
\begin{istgame}
\xtdistance{20mm}{20mm}
\istrooto[right](0)[oval node]<left>{player 1}
\istb \istb \endist
\istrooto(a)(0-1)<right>{player 2}+15mm..10mm+
\istb \istb \endist
\istrooto[right](b)(0-2)[box node]<135>{player 3}
\istb \istb \endist
\end{istgame}

Observe that the angle <180>, <left>, <right>, or <135> specifying the position of an owner’s name is redundant to \texttt{\istrooto}.

% Example 3: \istroot (counterclockwise)
\begin{istgame}
\xtdistance{15mm}{30mm}
\istrooto(0)[player 1]
\istb \istb \endist
\xtdistance{15mm}{15mm}
\istrooto(a)(0-1)<right>{player 2}
\istb \istb \endist
\istrooto(b)(0-2)[box node]{player 3}
\istb \istb \endist
\end{istgame}
The previous example shows how `\istrooto` arranges its branches: counterclockwise, by default.

### 4.2.2 `\istrooto'` – clockwise: swap version

The swap version of the oval version, `\istrooto'`, works just like `\istroot'` with one exception that it puts an owner within an oval node, by default.

The swap version is useful when a tree grows northwards or eastwards. The example below shows a tree rotated to the east by `\setistgrowdirection`. Note that `\istrooto'` arranges its branches clockwise.
Example 4: \istroot' (clockwise)
\begin{istgame}
\setistgrowdirection'{east}
\setistOvalNodeStyle{.6cm}
\istroot'(0){0}+{12.5mm}..{3.45cm}+
\istb \istb \endist
\xtShowEndPoints[oval node]
\xtdistance(12.5mm){11.5mm}
\istroot(1)(0-1){1}
\istb{}{3}[center] \istb{}{4}[center]
\endist
\istroot(2)(0-2){2}
\istb{}{5}[center] \istb{}{6}[center]
\endist
\istroot(6)(2-1){6}
\istb{}{9}[center] \istb{}{10}[center]
\endist
\draw[->,ultra thick,blue!20](260:1.5) arc (260:-80:1.5cm)
node [below,blue!30] {clockwise};
\end{istgame}

Remark: When you use the swap version \setistgrowdirection', using either \istroot or \istroot' makes no difference. (For more details, see Section 9 on page 45.)

5 Core macro: \istb

5.1 \istb: basics

5.1.1 Basics: branches, action labels, and payoffs

The macro \istb, basically, prints a branch. Having all arguments as options, a simple \istb draws a branch from a parent node designated by \istroot to a child node (or endpoint of \istb). If, for example, a parent node is named (A) by \istroot, the first child node is automatically named (A-1), the second child node (A-2), and so on.

The macro \istb also puts an action label and payoffs along with a branch, and does other functions. Note that the action labels and payoffs are to be typeset in math mode. If you want to change the input mode to text mode, you can use the macro \setistmathTF(*). This issue is discussed in Section 5.1.2 below. (You can also see Section 7, on page 39, for more details).

Remark: The macro \istb has many variants including \istb* (starred version), \istb (terminal version), and \istbm (missing version). It also has other very close friends: \istb (dual action label version) and \istbA (alternative or arc version). Each friend of \istb has its starred version.
and terminal version, of course, except \texttt{\textbackslash istbm}. \texttt{\textbackslash cntmistb(*)} and \texttt{\textbackslash cntmAistb(*)} are also friends only when a continuum of branches is in play.

branches

In the example below, each \texttt{\textbackslash istb} draws a branch. With the option \texttt{missing}, \texttt{\textbackslash istb} prints an invisible branch. Since the third child is \texttt{missing}, the last child is named \texttt{(0-4)}.

```
\begin{istgame}
\istroot (0)
  \istb \istb \istb<missing> \istb \endist
  \istb (D)(0-4)
  \istb \endist
\end{istgame}
```

To make it simple, you can use the \texttt{missing version} \texttt{\textbackslash istbm} instead of \texttt{\textbackslash istb<missing>}.

```
% \texttt{\textbackslash istbm}
\newcommand\istbm{\istb<missing>}
```

The macro \texttt{\textbackslash istb} also has various options to control the line style of a branch, and the direction and length of a branch. \texttt{\textbackslash istb} can also place payoffs to the direction of tree growth by default.

For various line styles of each branch, you can use any Ti\textrm{K}Z options of arrows, line style, color, and so on.

```
% Example: \texttt{\textbackslash istb} (branch line styles)
\begin{istgame}
\istroot (0)
  \istb[dashed, thick]
  \istb[->]
  \istb[draw=blue, thick] \endist
\end{istgame}
```

action labels

By default, an action label is put on the midpoint of the corresponding branch, in \texttt{math} mode.

```
% Example: action labels
\begin{istgame}
\istroot (0)
  \istb[dashed, thick]{A}
  \istb[->]{\beta}[right]
  \istb[draw=blue, thick]{Right}[ar] \endist
\end{istgame}
```

To specify the position of action labels, you can use the positional words or the abbreviations: \texttt{[a]} for \texttt{[above]}, \texttt{[b]} for \texttt{[below]}, \texttt{[l]} for \texttt{[left]}, \texttt{[r]} for \texttt{[right]}, \texttt{[al]} for \texttt{[above left]}, \texttt{[ar]} for \texttt{[above right]}, \texttt{[bl]} for \texttt{[below left]}, and \texttt{[br]} for \texttt{[below right]}.

\textbf{Remark:} Note that these abbreviations must be used with no other options, otherwise you will get a compile error. For example, when you want to print sloped labels for actions, you should do like \texttt{[above, sloped]} but not like \texttt{[a, sloped]}.
Warning: Issues in sloped labels in TikZ with asymmetric scales:

- The `tree` library in TikZ does not seem to treat sloped labels properly, when `xscale` or `yscale` is used asymmetrically.

This package provides a temporary solution to cure this issue: `\xtcureslopedlabelsNS` for trees growing northwards or southwards. If a tree grows eastwards or westwards, then use `\xtcureslopedlabelsEW`. These must be used in a TeX group with caution. These are only temporary solutions that have not been tested for every occasion.

payoffs

The second curly braces option of the macro `\istb` is for payoffs and the last bracket option for the position of payoffs.

By default, `\istb` prints payoffs in `math mode`, which can be changed by `\setistmathTF(*)`. By default, payoffs are put in the direction of `\istgrowdirection` ([south] by default).
Remark: What is \texttt{istgrowdirection} and what is it used for?

- \texttt{istgrowdirection} has the value of \texttt{[<grow keyval>]} typed in \texttt{istroot} (default: \texttt{south}).
- The value of \texttt{istgrowdirection} is (internally) used to determine the direction of putting payoffs (\texttt{south} by default).

In the following example, notice that the tree grows south-eastwards, so the payoffs are placed to the south-east of the endpoints.

\begin{verbatim}
\begin{istgame}
\istroot[-45](0)
\istb[dashed,thick]{A}{{(0,1)}}
\istb[-]{{(1,-1)}}
\istb[draw=blue,thick]{{\beta}}[a]{{u_1,u_2}}
\endist
\end{istgame}
\end{verbatim}

If you do not like the position of payoffs, you can change it by using degrees, the compass directions, or the positional words and their abbreviations mentioned above. In the example below, the tree grows south-westwards, but the position of payoffs at the end of the blue branch is changed to \texttt{[below]} or \texttt{[b]}.

\begin{verbatim}
\begin{istgame}
\istroot[south west](0)
\istb[dashed,thick]{A}{{(0,1)}}
\istb[-]{{(1,-1)}}
\istb[draw=blue,thick]{{\beta}}[r]{{u_1,u_2}[below]}
\endist
\end{istgame}
\end{verbatim}

Remark:

- \texttt{istb} expects a \textit{directional word} to be input as the last optional argument (by default, \texttt{istgrowdirection}).
- You can add more options to change the position of payoffs, but the first entry of the option list \textit{must} be a \textit{directional word}, like \texttt{[below,xshift=5mm]}, but not like \texttt{xshift=5mm,below}. Otherwise, a compile error will be produced.
- Note also that you can do like \texttt{[xshift=5mm]below}, instead (see also page 45).

5.1.2 Printing action labels in italics in text mode: \texttt{\setistmathTF*}

By default, an owner is printed in \textit{text mode}, action labels and payoffs in \textit{math mode}. With the macro \texttt{\setistmathTF(*)}, you can change the input mode for those labels. These are discussed in Section 7 on page 39.

Here, it is briefly discussed how to change the input mode of action labels to \textit{text mode}. To do that, just declare \texttt{\setistmathTF001}. The second zero means that the input mode for action labels is in text mode. (The first number is for owners and the third one for payoffs.)
If you use the starred version, like \setistmathTF*001, action labels are printed in italics, by default.

You can, of course, do the same thing in the default math mode.

5.2 \istb*: starred version

5.2.1 \istb*: basics

The starred version \istb* prints a solid node at the end of the corresponding branch. This is the only difference between \istb and \istb*.

5.2.2 \xtShowEndPoints and \xtHideEndPoints

Each endpoint is printed by each execution of \istb*. You can print solid nodes (by default) at all the endpoints of simple trees by the macro \xtShowEndPoints. You can also change the style of nodes for all the endpoints of simple trees, by specifying it as an optional argument, like \xtShowEndPoints[oval node].
The macro \texttt{\xtHideEndPoints} turns off the effects of \texttt{\xtShowEndPoints}.
It is too early to say that the starred version \texttt{\xtShowEndPoints*} additionally prints the two outermost endpoints of a continuum. \texttt{\xtHideEndPoints*} turns off only the outermost endpoints a continuum, but not the other endpoints. (See Section 11.4.3 on page 72, for more details.)

Here is an example of using \texttt{\xtShowEndPoints} and \texttt{\xtHideEndPoints}.

\begin{verbatim}
\xtShowEndPoints
\xtShowEndPoints[oval node,minimum size=6pt]
\istroot(0)[solid node]
\istb[dashed,thick]{A}
\istb
\istb[very thick]{\beta}[ar]
\endist
\xtShowEndPoints
\xtdistance{15mm}{10mm}
\istroot(b)(0-2)
\istb
\istb
\endist
\xtHideEndPoints
\xtHideEndPoints* \istb* overrides
\istroot(b)(0-3)
\istb
\istb*
\endist
\end{verbatim}

Note that \texttt{\xtShowEndPoints} and \texttt{\xtHideEndPoints} should be in an \texttt{istgame} environment to avoid unexpected results. Note also that \texttt{\istb*} overrides all the effects of these two macros by forcing to print a solid node.

5.3 \texttt{\istbt}: terminal version

5.3.1 \texttt{\istbt}: basics

The terminal version \texttt{\istbt} is designed to represent a terminal move in a game tree. Basically, \texttt{\istbt} works exactly the same way as \texttt{\istb} does. However, using \texttt{\istbt} together with the macro \texttt{\xtShowTerminalNodes} you can control the shape of the terminal nodes, all at once.

5.3.2 \texttt{\xtShowTerminalNodes} and \texttt{\xtHideTerminalNodes}

The terminal version \texttt{\istbt} used with \texttt{\xtShowTerminalNodes} prints a solid node (by default). You can change the style of the terminal nodes, like \texttt{\xtShowTerminalNodes[oval node]}. This effect can be turned off by \texttt{\xtHideTerminalNodes}.
Remark: Note that controlling terminal nodes by using \texttt{\textbackslash xtShowTerminalNodes} works only with the terminal versions \texttt{\textbackslash istbt}, \texttt{\textbackslash istBt} (documented below in Section 5.4) and \texttt{\textbackslash istbtAt} (in Section 5.5), but there will be no effect and no harm with other versions. Note also that \texttt{\textbackslash istbt*} overrides the effects of \texttt{\textbackslash xtShowTerminalNodes} and \texttt{\textbackslash xtHideTerminalNodes}.

5.4 \texttt{\textbackslash istB}: dual action label version

5.4.1 \texttt{\textbackslash istB}: basics

The macro \texttt{\textbackslash istB} works just like \texttt{\textbackslash istb}, except one thing: \texttt{\textbackslash istB} prints dual action labels for a branch.

The starred version \texttt{\textbackslash istB*} prints a solid node at the end of the corresponding branch, just like \texttt{\textbackslash istb*}. And the terminal versions \texttt{\textbackslash istBt} and \texttt{\textbackslash istBt*} work just like \texttt{\textbackslash istbt} and \texttt{\textbackslash istbt*}, respectively, except for dual action labels.

With \texttt{\textbackslash istB}, do not forget to put \texttt{two labels}. Otherwise, you might get an unexpected result.
5.4.2 \texttt{\textbackslash xtActionLabel} and \texttt{\textbackslash xtActionLabel*}

You can also use the supplementary macro \texttt{\textbackslash xtActionLabel} to print additional action labels (see also Section 12.2 on page 78). The macro \texttt{\textbackslash xtActionLabel} controls, from outside of a simple tree, a branch (by default \texttt{\textbackslash draw=none}) with a label (in math mode by default), connecting two nodes.

\begin{verbatim}
% \texttt{\textbackslash xtActionLabel}
% syntax:
% \texttt{\textbackslash xtActionLabel*}[<line opt>]<from><to>{<action>}[<pos>,<node opt>]
% default:
% [-,draw=none]<mm><mm>{black,text depth=.25}
\end{verbatim}

Remark: In some cases, it is not a good idea to use \texttt{\textbackslash istB} to print dual action labels.

- With \texttt{\textbackslash istB}, the features of \textit{middle arrows} provided by this package does not work well (see Section 12.2 on page 78).
- With respect to a \textit{continuum} of branches, \texttt{\textbackslash istB} has little role to play.

You can do the same thing as \texttt{\textbackslash istB} does using \texttt{\textbackslash istb} together with \texttt{\textbackslash xtActionLabel}.

\begin{verbatim}
% \texttt{\textbackslash istb and \textbackslash xtActionLabel}
\begin{istgame}[font=\footnotesize]
\xtdistance{15mm}{30mm}
\istroot(0)[chance node]
\istb[Left][al]
\istb[Right][ar]
\endist
\istroot(1)(0-1)[initial node]
\istb{Head}[al]
\istb{Tail}[ar]
\endist
\xtActionLabel(0)(0-1){x}[br]
\xtActionLabel(0)(0-2){1-x}[bl]
\xtActionLabel(1)(1-1){y}[br]
\xtActionLabel[draw,blue,thick](1)(1-2){1-y}[bl]
\end{istgame}
\end{verbatim}

The starred version \texttt{\textbackslash xtActionLabel*} prints its label in the input mode as set by \texttt{\textbackslash setistmathTF*} (see Section 7 on page 39, for more details on \texttt{\textbackslash setistmathTF}).

\begin{verbatim}
% \texttt{\textbackslash istb and \textbackslash xtActionLabel*}
\begin{istgame}[font=\footnotesize]
\xtdistance{15mm}{30mm}
\istroot(0)[chance node]
\istb[Left][al]
\istb[Right][ar]
\endist
\istroot(1)(0-1)[initial node]
\istb{Head}[al]
\istb{Tail}[ar]
\endist
\xtActionLabel(0)(0-1){x}[br]
\xtActionLabel(0)(0-2){1-x}[bl]
\xtActionLabel*{(1-1){y}[br]
\xtActionLabel*[draw,blue,thick](1)(1-2){1-y}[bl]
\end{istgame}
\end{verbatim}
5.5 \texttt{\textbackslash istb\textbackslash A}: alternative (or arc) version

5.5.1 \texttt{\textbackslash istb\textbackslash A}: basics

The macro \texttt{\textbackslash istb\textbackslash A} is an alternative (or arc) version, doing one more thing than \texttt{\textbackslash istb}. With \texttt{\textbackslash istb\textbackslash A} you can easily change the level distance of an individual branch using a factor (1, by default) as the first optional argument in parentheses. All other arguments are the same as in \texttt{\textbackslash istb}. For example, \texttt{\textbackslash istb\textbackslash A(.5)} is an abbreviation of \texttt{\textbackslash istb<level distance=.5*\xtlevdist>}, as you can see in the example below:

\begin{verbatim}
\% \texttt{\textbackslash istb\textbackslash A (alternative or arc version)}
\begin{istgame}
\istroot(0) \istb \istb\textbackslash A \endist
\istroot(1)(0-1) \istb \istb\textbackslash A(.5){b} \endist
\istroot(2)(0-2) \istb
\istb\textbackslash A<level distance=.5*\xtlevdist>{b} \endist
\end{istgame}
\end{verbatim}

You can interchangeably use \texttt{\textbackslash istb\textbackslash A} and \texttt{\textbackslash istb}, except for one case, in which you are working with a continuum of branches.

Remark: (too early to comment)

- The macro \texttt{\textbackslash istb\textbackslash A} is originally created to work with \texttt{\textbackslash istrootcntmA} as an arc version.
- \texttt{\textbackslash istb\textbackslash A} draws, by default, a branch up to an arc when used with the continuum arc version \texttt{\textbackslash istrootcntmA}, but \texttt{\textbackslash istb} does not. (For more details, see Section 11.2.1 on page 65.)
- Except for the case of using a continuum of branches, \texttt{\textbackslash istb\textbackslash A} is equivalent to \texttt{\textbackslash istb} for users.

Its terminal version \texttt{\textbackslash istb\textbackslash At} is also available, but the dual action label version of \texttt{\textbackslash istb\textbackslash A} is not provided. The starred versions \texttt{\textbackslash istb\textbackslash A*} and \texttt{\textbackslash istb\textbackslash At*} print a solid node (by default) at the end of the corresponding branch.

5.5.2 \texttt{\textbackslash istb}: applications

By specifying the \texttt{\textbackslash grow} key of \texttt{\textbackslash istb} you can draw a branch with the exact length you want.

\begin{verbatim}
\% Example: \texttt{\textbackslash istb\textbackslash A}
\begin{istgame}
\xtdistance{10mm}{15mm}
\istroot(0) \istb\textbackslash A(.2) \istb\textbackslash A(1.5) \istb\textbackslash A(1.2) \endist
\istroot(1)(0-1)
\istb\textbackslash A(.7)<grow=-120> \istb
\istb\textbackslash A(1)<grow=-30>{b} \endist
\istroot(2)(0-3) \istb \istb\textbackslash A\{b\} \endist
\draw[dashed] (1) circle (10mm);
\draw[dotted] (1) circle (7mm);
\end{istgame}
\end{verbatim}

You can also apply \texttt{\textbackslash istb\textbackslash A} to easily ruin your regular balanced trees.
6 Important labels: players, action labels, and payoffs

6.1 How to put players

6.1.1 Players: basics

The macro \texttt{\textbackslash istroot} specifies the (sub)root of a simple tree and puts its owner (or a player). The direction to which a player label is put is set by the angle \texttt{<\textit{angle}>} option of \texttt{\textbackslash istroot}, like \texttt{<above>} (by default), \texttt{<east>}, or \texttt{<45>}. To specify the direction you can use degrees, the compass directions, or positional words.

\texttt{% Example: node owner}
\begin{itemize}
\item \begin{itemize}
  \item \texttt{\begin{istgame}}
  \texttt{\istroot(0)<above>{Child} \ % default: <above>}
  \texttt{\istb \istb \endist}
  \texttt{\istroot(1)(0-2)<45>{Parent}}
  \texttt{\istb \istb \endist}
  \texttt{\end{istgame}}
\end{itemize}
\end{itemize}

\texttt{\% Example: node owner (with \texttt{\textbackslash istroot})}
\begin{itemize}
\item \begin{itemize}
  \item \texttt{\begin{istgame}}
  \texttt{\istrooto(0){Child}}
  \texttt{\istb \istb \endist}
  \texttt{\istrooto(1)(0-2)<45}\texttt{\{\textbf{Parent}}}
  \texttt{\istb \istb \endist}
  \texttt{\end{istgame}}
\end{itemize}
\end{itemize}

Remark:

- The supplementary macro \texttt{\textbackslash xtOwner} is provided as an extra way of putting owners of decision nodes (see Section 13.1 on page 83).
- Though an owner is to be input in text mode, by default, it is also possible to change the input mode to math mode with \texttt{\setistmathTF(*)} (see below in Section 6.1.4 and Section 7 on page 39).

Note that the oval version \texttt{\textbackslash istrooto} produces a bubble type node with an owner in it, so the directional option \texttt{<\textit{angle}>} is redundant with \texttt{\textbackslash istrooto} (see Section 4.2 on page 19).

\texttt{% Example: coloring players}
\begin{itemize}
\item \begin{itemize}
  \item \texttt{\begin{istgame}}
  \texttt{\istroot(0)<[red]>{Child}}
  \texttt{\istb \istb \endist}
  \texttt{\istroot(1)(0-2)<[blue]45>{\textsf{Parent}}}
  \texttt{\istb \istb \endist}
  \texttt{\end{istgame}}
\end{itemize}
\end{itemize}

6.1.2 Coloring players or a whole simple tree using \texttt{\textbackslash istroot}

You can change the color of a player’s name by giving \texttt{color} in the angle option of \texttt{\textbackslash istroot}, like \texttt{<[red]>} or \texttt{<[blue]45>}. (This is the \LaTeX\ way of giving options for the \texttt{label} nodes.)
The example below shows how you color a whole simple tree: red for the Child’s simple tree and blue for the Parent’s simple tree.

% Example: coloring a simple tree
\begin{istgame}
\istroot[south,red](0) % whole simple tree, but node
\istb{l}{0}[south]
\istb{r}{2}[south]
\endist
\istroot[south,draw=blue](1)(0-2)<45>{\tttexttt{Parent}} % for owner only
\istb{l}{1}[1,-1][south]
\istb{r}{1}[1,1][-90]
\endist
\end{istgame}

Remark:

- In the above example, the first \istroot has three bracket [option]'s: the first bracket option is for a whole simple tree including branches and labels (except a node style), the second one is for a node style of a decision node, and the third one is for the owner of the (sub)root.
  
  - If you want to have all the branches, action labels, and payoffs in red, just use the color name like \texttt{[south, red]}. If you want to have just the branches in blue, use the \texttt{draw} option key like \texttt{[south, draw=blue].} (This is what TikZ does.)

- Note that the first entry in the first [option] list of \istroot must be the growing direction of a simple tree. So it should be input like \texttt{[south, red]}, but never like \texttt{[red, south].} (This is what \istroot requires.) Otherwise, you will get a compile error.

6.1.3 Decorating players or a whole simple tree using \istrooto

The oval version \istrooto puts an owner within a bubble type node. With \istrooto, the color of a player can be changed by the second bracket [option], such as \texttt{[red]} or \texttt{[blue].} (This is the TikZ way for giving options for the main nodes.)

% Example: decorating players (with \istrooto)
\begin{istgame}
\istrooto[below,red](0)\{Child\}
\istb{a}{B}[below] % for node only
\istb{c} \endist
\istrooto[below,draw=blue](1)(0-2)<45>{\tttexttt{Parent}} % for owner only
\istb{a}{B}[below] % for node only
\istb{c} \endist
\end{istgame}

The following example shows how you can color, with \istrooto, a whole simple tree including action labels and payoffs in red or blue, except for node styles.

% Example: decorating players (with \istrooto)
\begin{istgame}
\istrooto[south,red](0)[draw=green, text=blue]\{Child\}
\istb*{a}{B}[below] % for node only
\istb*{c} \endist
\istrooto[-90,blue](1)(0-2)[draw=blue, text=blue]
\rightarrow \{Parent\}
\istb* \endist
\end{istgame}
What if you want to paint some color into the background of each oval node or box node? You can also do this by simply using the TikZ way of giving options for nodes.

\begin{istgame}
\istroot(0){Child}\istb\endist
\istroot(1)(0-2)[box node,fill=red!20,text=blue]{Parent}\istb\endist
\end{istgame}

6.1.4 Changing the input mode and text font style: $\setistmathTF(\ast)$

You can change the input mode for owners from text mode (by default) to math mode by using the macro $\setistmathTF111$ which has three mandatory arguments. The first number 1 means that an owner is printed in math mode. (The second number 1 and third number 1 mean that action labels and payoffs are in math mode, respectively.) $\setistmathTF011$ means that an owner is printed in text mode (this is default).

\begin{istgame}
\setistmathTF111 \% owner: math mode
\istroot(0){\Omega_{\alpha}}+10mm..20mm+ \istb \endist
\setistmathTF011 \% owner: text mode
\istroot(1)(0-1)<180>{Alpha 2} \istb \endist
\end{istgame}

The starred version, for example, $\setistmathTF\ast011$, enables you to change the font style of an owner. Moreover, by specifying a text font style, like $\setistmathTF\ast011<texttt>$, in angle brackets, you can print an owner in typewriter font. (See Section 7 on page 39, for more details on $\setistmathTF\ast$.)

\begin{istgame}
\setistmathTF111 \% owner: math mode
\istroot(0){\Omega_{\alpha}}+10mm..20mm+ \istb \endist
\setistmathTF011 \% owner: text mode
\istroot(1)(0-1)<180>{Alpha 2} \istb \endist
\setistmathTF\ast011<texttt> \% owner: in texttt
\istroot(2)(0-2)<0>{Omega 2} \istb \endist
\end{istgame}

The supplementary macro $\xtOwner$ gives an alternative way of putting, outside of a simple tree, an owner in text mode at a node. The starred version $\xtOwner\ast$ prints an owner as in the input mode set by $\setistmathTF\ast$. (See Section 13.1 on page 83, for more details on $\xtOwner$.)
6.2 How to put action labels

6.2.1 Action labels: basics

The macro \texttt{\istb} prints a branch and its action label. Note that action labels should be input in \textit{math mode}, by default.

By default, \texttt{\istb} prints its action label on the midpoint of the corresponding branch. You can specify the position of an action label with the positional words and abbreviations but not by the compass directions or degrees.

In the following example, the abbreviations are used to place action labels.

Notice that with the \textit{abbreviations} the position of an action label is (internally) adjusted to get better result (for more details, see Section 8.2.2 on page 44).

Remark:

- The supplementary macro \texttt{\xtActionLabel} is provided as an alternative way to put action labels (see Section 13.1 on page 83, for more details).
- By default, action labels are to be input in \textit{math mode}. You can change the input mode to text mode by using \texttt{\setistmathTF} or \texttt{\setistmathTF*} (see below in Section 6.2.5 and Section 7 on page 39).

33
6.2.2 Decorating action labels

You can change the color of action labels with TiKZ options. In the example below, the bracket options before an action label are for branches and those after are only for the action labels.

% Example: color and line style
\begin{istgame}
\istroot(0)
\istb[dashed,thick]{A}[al]
\istb[blue,very thick]{B}[r]
\istb[blue,very thick]{C}[above right,red]
\endist
\end{istgame}

Remark: It is important to remember that you cannot use the abbreviations with additional options. For example, you can do like \istb{C}[ar] but not like \istb{B}[ar,red]. Instead, you should do like \istb{C}[above right,red].

You can also express action labels in a box or a circle, or other shapes with colors.

% Example: decorating action labels
\begin{istgame}
\istroot(0)
\istb{A}[above left,draw=black,circle,fill=red!20]
\istb[B][right,draw=black,fill=green,inner sep=0pt]
\istb[\textbf{C}][above right,fill=blue,text=yellow]
\endist
\istroot(1)(0-2)
\istb{D}[above left,xshift=-3pt,draw=blue,double]
\istb{E}[above right,xshift=5pt,draw,star]
\endist
\end{istgame}

6.2.3 Sloped action labels

You can print sloped labels for actions by using the TiKZ option sloped. Still you should remember that you cannot use the abbreviations to place action labels with other options like sloped.

% Example: sloped action labels
\begin{istgame}
\istroot(0)
\istb{Good}[above,sloped]
\istb{Bad}[above,sloped]
\endist
\istroot(1)(0-2)
\istb[\text{\fbox{$\alpha$}}][above,sloped]
\istb[\text{\fbox{$\beta$}}][above,sloped]
\endist
\end{istgame}

Warning: Issues in sloped labels in TiKZ with asymmetric scales:

- The tree library in TiKZ does not seem to treat sloped labels properly, when xscale or yscale is used asymmetrically.
A temporary solution to cure this issue is to declare the macro `\xtcureslopedlabelsNS`, provided with this package, in a TeX group with caution. For a tree growing eastwards or westwards, use `\xtcureslopedlabelsEW`, instead. (These solutions are only temporary. They have not been tested for every occasion.)

6.2.4 Dual action labels

You can also express dual action labels for actions by using `\istB` or `\istBt`.

You can do the same thing as `\istB` does by using `\istb` together with the supplementary macro `\xtActionLabel` (see Section 5.4.2 for more details on dual action labels).

6.2.5 Changing the input mode and text font style: `\setistmathTF(*)`

By default the input mode for action labels is in math mode. With `\setistmathTF`, you can change the input mode for labels. The package’s default input mode is set as `\setistmathTF011`, meaning that the input mode for an owner is in text mode (denoted by the first 0), for action labels in math mode (denoted by the second number 1), and for payoffs in math mode (denoted by the third number 1).

So if you do like `\setistmathTF001` (with the second number 0), action labels are in text mode.
Moreover, if you use the starred version, like \setistmathTF*001, action labels automatically are printed in *italics*. If you specify a font style in *curly braces*, like \setistmathTF*001{texttt}, you can even print action labels in typewriter font. For more details, see Section 7 on page 39.

The supplementary macro \xtActionLabel prints, from outside of a simple tree, an action label in *math mode*. The starred version \xtActionLabel* prints an action label in the input mode as set by \setistmathTF*. For more details, see Section 5.4.2 (on page 28) and Section 7 (on page 39).

### 6.3 How to put payoffs

#### 6.3.1 Payoffs: basics

The macro \istb can print a branch and also the corresponding payoffs, in *math mode* by default, near at its endpoint. The payoffs are put in the direction set by \setistgrowdirection (south, by default), unless it is changed by <grow keyval> of \istroot.

Remark:

- The supplementary macro \xtPayoff is provided for an extra way of putting payoffs (see Section 13.1 on page 83).
- The payoffs are to be input in math mode, by default, you change the input mode to text mode by using \setistmathTF(*) (see Section 7 on page 39).
6.3.2 Payoffs and `\texttt{\textbackslash istgrowdirection}`

The direction of where payoffs are put from a terminal node follows `\texttt{\textbackslash istgrowdirection}` typed in as the first optional argument of `\texttt{\textbackslash istroot}`. The default direction is `south` and can be changed by `\texttt{\textcolor{blue}{\textbackslash setistgrowdirection}}`. For example, `\texttt{\textcolor{blue}{\textbackslash setistgrowdirection\{north\}}}` changes the default direction to `north`.

To specify the tree growing direction or the position of payoffs to be put, you can use degrees, the compass directions, or the positional words and their abbreviations.

\begin{quote}
\verb|\begin{istgame}|
\verb|\istroot(0)|
\verb|  \istb[dashed,thick]{A}{\binom{2}{3}}|
\verb|  \istb*[{}{(-2,3)}]|
\verb|  \istbt[blue,very thick]{B}[ar]{2,3}|
\verb|\endist|
\verb|\end{istgame}|
\end{quote}

By default, `grow=south`, so `\texttt{\textbackslash istgrowdirection}` is `south` (or `below` or `-90`). The above example shows payoffs at the south (by default) of terminal nodes.

**Remark:**

- You can simply omit the position of payoffs. Then they are printed, by default, in the direction as set by `\texttt{\textcolor{blue}{\textbackslash istgrowdirection}}`.
- However, if you use options other than the direction of payoffs, you must specify the direction with others.
- Moreover, the direction must be the first entry of the option list for payoffs and the abbreviations cannot be used. For example, `[b,yshift=-3mm]` and `[yshift=-3mm,below]` are not acceptable. It must be that `[below,yshift=-3mm]`.
- Note also that you can do like `[[yshift=-3mm]below]`, instead (see also page 45).

\begin{quote}
\verb|\begin{istgame}|
\verb|\istroot(0)|
\verb|  \istb[dashed,thick]{A}{\binom{2}{3}}|
\verb|  \istb*[{}{(-2,3)}]|
\verb|  \istbt[blue,very thick]{B}[ar]{2,3}|
\verb|    [below,yshift=-3mm,draw]|
\verb|\endist|
\verb|\end{istgame}|
\end{quote}

You can see more examples, below, that show the positions of payoffs, by default, depend on the tree growing directions, unless you specify different directions.

\begin{quote}
\verb|\begin{istgame}|
\verb|\istroot[right](0)|
\verb|  \istb[dashed,thick]{A}{\binom{2}{3}}|
\verb|  \istb*[{}{(-2,3)}]|
\verb|  \istbt[blue,very thick]{B}[al]{2,3}|
\verb|\endist|
\verb|\xtHideTerminalNodes|
\verb|\end{istgame}|
\end{quote}

grow=right=`\texttt{\textcolor{blue}{\textbackslash istgrowdirection}}`, so payoffs are put on the right.
grow=north=\texttt{\textbackslash istgrowdirection}, so payoffs are put above the terminal nodes.

\begin{itemize}
  \item grow=south west=\texttt{\textbackslash istgrowdirection}, so payoffs are put below left of the terminal node.
\end{itemize}

**Remark:** You can adjust the direction of putting payoffs by specifying a directional word right after payoffs, like \texttt{\textbackslash istb[blue, very thick]\{B\}[right]{2,3}[below]}.

**Remark:** You can use the abbreviations [l], [r], [a], and [b] for [left], [right], [above], and [below], respectively. The abbreviations [al], [ar], [bl], and [br] can also be used for [above left], [above right], [below left], and [below right], respectively, to put payoffs (for more details about abbreviations, see Section 8.2.2 on page 44).

**Remark:** Notice also that, instead of the positional words, you can use the compass directions or degrees, like \texttt{\textbackslash istb[blue, very thick]\{B\}[right]{2,3}[-90]}.  

### 6.3.3 Decorating payoffs

You can change the color of payoffs by giving Ti\textsc{latex} options right before the positional words for payoffs. For example you can do like \texttt{\textbackslash istb...\{(1,1)\}[[blue]below]. Note that, in this case, you cannot use the abbreviation of the positional words.

You can also put payoffs in a box, a circle, or other shapes, even with color in the background.
6.3.4 Changing the input mode and text font style: \texttt{\setistmathTF(*)}

You can change the input mode for payoffs from math mode (by default) to text mode, by using \texttt{\setistmathTF010}. The last 0 means that payoffs are in text mode.

The starred version, for example, \texttt{\setistmathTF*010[texttt]}, with a font style specified in brackets, prints payoffs in typewriter font.

The supplementary macro \texttt{\xtPayoff} prints, outside of a simple tree, payoffs in \textit{math mode}. The starred version \texttt{\xtPayoff*} prints payoffs in the input mode as set by \texttt{\setistmathTF*}. See Section 7 on page 39.

7 Input mode and text font style changer: \texttt{\setistmathTF(*)}

The macro \texttt{\setistmathTF} enables you to change the input mode for important labels: owners, action labels, and payoff. It takes three numbers (0 or 1) as \textit{mandatory} arguments. Here, 1 (true) means the input mode is in \textit{math mode} and 0 (false) in \textit{text mode}. The three numbers represents
the input mode for owners, action labels, and payoffs, respectively. In this package, it is initially set as \texttt{\setistmathTF011}, meaning that owners are to be input in text mode, and action labels and payoffs in math mode.

% \setistmathTF
% syntax:
% \setistmathTF\{<owner input mode>\}\{<action input mode>\}\{<payoff input mode>\}
% initial values:
% \{0\}\{1\}\{1\}

The starred version \texttt{\setistmathTF*} accepts (in addition to three mandatory numbers) three optional arguments, each of which is effective only when the corresponding input mode is in text mode. The arguments must be one of text font styles \textit{without a backslash} such as \texttt{textrm, textit, itshape, textbf, scriptsize, tiny}, and so on. Each of the three optional arguments is ignored when used with \texttt{\setistmathTF} and when the corresponding input mode is in math mode.

% \setistmathTF
% syntax:
% \setistmathTF\{\}\{\}\{<owner font>\}\{<action label font>\}\{<payoff font>\}
% defaults
% \{0\}\{1\}\{1\}<>\{\textit\}\{}

The first optional argument in \textit{angle brackets} is the font style for an owner, the second in \textit{curly braces} for action labels, and the third \textit{in brackets} for payoffs. By default action labels are printed in italics.

For example, with \texttt{\setistmathTF\*000\texttt{\textit\}[\texttt{tiny}]} declared, an owner is printed in typewriter font, action labels in italics, and payoffs in the tiny size of normal text font (roman, upright). With \texttt{\setistmathTF\*000\texttt{\textit\}[\texttt{tiny}]} all the optional arguments are ignored and all the labels are printed in normal text font. Note also that with \texttt{\setistmathTF\*001}, you can print action labels in italics (by default).

7.1 \texttt{\setistmathTF}: input mode changer

The macro \texttt{\setistmathTF} is an \textit{input mode changer}, taking three numbers as mandatory arguments. The default input modes for important labels are set as \texttt{\setistmathTF011}.

% \setistmathTF
\begin{istgame}
%\setistmathTF011 % (default mode)
\xtdistance{20mm}{20mm}
\istroot(0){Alan 1}+20mm..40mm+
  \istb{left 1}[al]
  \istb{right 1}[ar] \endist
\setistmathTF001 % mode: text,text,math
\istroot(1)(0-1)<180>{Bob 2}
  \istb{left 2}[al]{pie 1}
  \istb{right 2}[ar]{pie 2} \endist
\setistmathTF100 % mode: math,text,text
\istroot(2)(0-2)<0>{Kim 3}
  \istb{left 3}[al]{pie 3}
  \istb{right 3}[ar]{pie 4} \endist
\end{istgame}
7.2 \texttt{\setistmathTF*: input mode and text font style changer}

The starred version \texttt{\setistmathTF*} is a \textit{text font style changer} as well as an \textit{input mode changer}. It takes three numbers as \textit{mandatory} arguments followed by three \textit{optional} arguments in the order of \texttt{<>{}[]} . The first option \texttt{<>} is for an owner, the second \texttt{[]} for action labels, and the third \texttt{[]} for payoffs.

Each optional argument should be one of valid font shapes and sizes, but without a textbackslash (not as a command), such as \texttt{textbf}, \texttt{textsc}, \texttt{textit}, \texttt{large}, \texttt{tiny}, and so on. Each of the options is effective only when it is used with the starred version \texttt{\setistmathTF*} and the corresponding input mode is in text mode.

Remark: Note that if the second optional argument is omitted when action labels are in text mode like, for example, \texttt{\setistmathTF*001}, action labels are printed in \textit{italics} by default.

\begin{lstlisting}[language=TeX]
% \setistmathTF*
\begin{istgame}
\xtdistance{20mm}{20mm}
\setistmathTF*011<textsc}{tiny}[textbf]
\istroot(0){(Alan 1)+20mm..40mm+} \istb(left 1)[al]
\istb(right 1)[ar] \endist
\setistmathTF*001<textsc>{tiny}[textbf]
\istroot(1)(0-1)<180>{(Bob 2)} \istb(left 2)[al]{pie 1}
\istb(right 2)[ar]{pie 2} \endist
\setistmathTF*100[\textbf]
\xtOwner*(0){Alan 1} \xtOwner(1){Bob 2} \xtActionLabel*(0)(0-1){left 1}[al]
\xtActionLabel(1)(1-1){left 2}[al]
\xtPayoff*(1-1){pie 1}
\end{istgame}
\end{lstlisting}

7.3 \texttt{\setistmathTF*} and supplementary macros \texttt{\xtFoo*} printing labels

Many supplementary macros, working outside of a simple tree, optionally print important labels: owners, action labels, and payoffs. These include \texttt{\xtOwner}, \texttt{\xtActionLabel}, \texttt{\xtPayoff}, \texttt{\xtInfosetOwner} and many more. All of the macros for information sets (except \texttt{\cntmAInfoset} and \texttt{\txtmAInfosetO}) such as \texttt{\xtInfoset} or \texttt{\xtCInfosetO} also print the owners of information sets. With any supplementary macros, by default, an owner is printed in text mode and action labels and payoffs are printed in math mode.

All of these macros have their own \textit{starred} (\*) versions, which print the labels in the input mode as set by \texttt{\setistmathTF*} (but not by \texttt{\setistmathTF}). For example, \texttt{\xtActionLabel*} prints action labels in \textit{italics} with \texttt{\setistmathTF*001}, while \texttt{\xtActionLabel} prints action labels in \textit{math mode}.

\begin{lstlisting}[language=TeX]
% Example: \setistmathTF* and \xtActionLabel*
\begin{istgame}
\xtdistance{20mm}{20mm}
\setistmathTF*000<textsc}{tiny}[textbf]
\istroot(0)+20mm..40mm+ \istb \istb \endist
\istroot(1)(0-1) \istb \istb \endist
\xtOwner*(0){(Alan 1)} \xtOwner(1){(Bob 2)} \xtActionLabel(0)(0-1){left 1}[al]
\xtActionLabel(1)(1-1){left 2}[al]
\xtPayoff*(1-1){pie 1}
\end{istgame}
\end{lstlisting}
In this regard, you can find more examples of supplementary macros scattered here and there in appropriate places throughout the manual. See Section 6.1.4 Section 6.2.5 Section 6.3.4.

Here are some more examples on owners of information sets (see Section 10 for more details on information sets).

\begin{istgame}
\setistmathTF*011\textscf\texttt\textbf
\istroot(0){Alice}+15mm..30mm+ \istb(A) \istb(B) \endist
\istroot(1)(0-1)<135>{Ben} \istb(C) \istb(D) \endist
\istroot(2)(0-2) \istb(a) \istb(b) \endist
\istroot(3)(1-2) \istb(a) \istb(b) \endist
\setistmathTF*011\textscf\texttt\textbf
\xtInfosetO(0)(0)
\xtInfosetO[rectangle,rounded corners=.2em](1)(1)
\xtInfosetO*[ellipse,fill=blue!60]((3)(2){Cate}[sloped,white](1.5em)
\end{istgame}

\begin{istgame}
\setistmathTF*100\textscf\texttt
\istroot(0){\alpha}+15mm..30mm+
\istb{Left}[a] \istb{Right}[a]\endist
\istroot(1)(0-1) \istb \istb \endist
\istroot(2)(0-2) \istb \istb \endist
\xtdistance{10mm}{10mm}
\istroot(3)(1-1) \istb \istb \endist
\istroot(4)(1-2) \istb \istb \endist
\istroot(5)(2-1) \istb \istb \endist
\istroot(6)(2-2) \istb \istb \endist
\istb \istb[Go][r]{omega} \endist
\xtInfoset*(1)(2)
\xtInfosetOwner*(1)(2){beta}[a]
\setistmathTF*011\textscf\texttt
\xtCInfoset*(1-1)(2-1){Ben}
\xtCInfoset*(1-2)(2-2){Cate}
\end{istgame}

\begin{istgame}
\setistmathTF*011\texttt\textbf
\istroot(0){Alice}+15mm..30mm+
\istb \istb \endist
\istroot(1)(0-1) \istb \istb \endist
\istroot(2)(0-2) \istb \istb \endist
\xtdistance{10mm}{10mm}
\istroot(3)(1-1) \istb \istb \endist
\istroot(4)(1-2) \istb \istb \endist
\istroot(5)(2-1) \istb \istb \endist
\istroot(6)(2-2) \istb \istb \endist
\xtCInfosetO*(1)(0){imperfect recall}[l]
\xtCInfosetO*[dashed,blue,thick]
(1)(2)<.7>{Blue}
\xtCInfosetO(1-1)(2-1){Ben}
\xtCInfosetO(1-2)(2-2){Cate}
\end{istgame}
8 Fine-tuning positions of players, action labels, and payoffs (experimental)

8.1 Fine-tuning positions: owners

If you are not satisfied the position of an owner (or a player), you can change it by using the TikZ options such as xshift, yshift, or label distance with the angle option of \istroot.

Examples are \{xshift=10pt\}above\{Child\} and \{label distance=-5pt\}45\{Parent\}, as shown in the following:

\begin{lstlisting}[language={TikZ}, framex=W]{istgame}
\istroot(0)<[xshift=10pt]above>{Child}
\istb
\istb
\endist\istroot(1)(0-2)<[label distance=-5pt]45>{Parent}
\istb
\istb
\endist\end{istgame}

8.2 Fine-tuning positions: action labels

8.2.1 Abbreviations: \[l\], \[r\], \[a\], and \[b\]

As discussed in 5.1 on page 21, the macro \istb deals with the action labels.

In order to determine the direction of action labels for branches to put, you can use degrees, the compass directions, or the positional words and the abbreviations as mentioned above. (Internally, the abbreviations for payoffs and those for action labels work slightly differently in terms of xshift and yshift.)

\begin{lstlisting}[language={TikZ}, framex=W]{istgame} \begin{istgame}[scale=1.2]
\xtShowEndPoints \xtdistance{12mm}{16mm}
\istroot(0)[initial node]
\istb<grow=0>{\fbox{$a$}}[a] \istb<grow=90>{\fbox{$b$}}[l]
\istb<grow=180>{\fbox{$\$a$}}[b] \istb<grow=-90>{\fbox{$\$b$}}[r]
\endist\end{istgame}

When you use these abbreviations you can manipulate the horizontal and/or the vertical shifts toward branches by using \xtALPush. (This is experimental!)

% syntax:
\xtALPush{xshift dim} for \[l\] and \[r\]{yshift dim} for \[a\] and \[b\]
% default:
{0pt}{0pt}

For example, \xtALPush{-3pt}{5pt} draws the labels left and right by 3pt to the branch and push those put above and below 5pt away from the branch, as shown in the following:
8.2.2 Abbreviations: \[al\], \[ar\], \[bl\], and \[br\]

You can also use the abbreviations \[al\], \[ar\], \[bl\], and \[br\] to represent above left, above right, below left, and below right, respectively, to position action labels. Precise representation of abbreviations is as follows:

- \[al\] represents \[above left, xshift=1pt, yshift=-2pt, black\]
- \[ar\] represents \[above right, xshift=-1pt, yshift=-2pt, black\]
- \[bl\] represents \[below left, xshift=1pt, yshift=2pt, black\]
- \[br\] represents \[below right, xshift=-1pt, yshift=2pt, black\]

\begin{figure}
\centering
\begin{tikzpicture}[scale=1.2, every node/.style={font={\small}}]
\begin{scope}
\node at (0,0) (node) {0};
\end{scope}
\end{tikzpicture}
\caption{Positioning action labels with the abbreviations}
\end{figure}
In Figure 1, \texttt{al}'s are put in the same position from dotted branches, and so are the other labels from their corresponding branches, like \texttt{bl}'s from the blue dashed branches.

You can also use \texttt{\textbackslash xtALShift} to put and push labels horizontally and vertically. (This is experimental!)

% syntax:
\texttt{\textbackslash xtALShift}\{<horizontal shift dim>\}\{<vertical shift dim>\}

% defaults:
\{1pt\}\{2pt\}

When the dimensions get bigger than the defaults (1pt and 2pt) the labels get closer to the midpoints of the corresponding branches, and when the numbers get smaller the labels get farther from their branches.

% Example: node owner
\begin{istgame}
\xtALShift\{4pt\}\{3pt\}
\istroot\{0\}\{Child\} % default: <above>
\istb\{A\}\{al\} \istb\{B\}\{ar\} \endist
\istroot\{1\}\{0-2\}\{<45\}\{Parent\}
\istb\{L\}\{al\} \istb\{R\}\{ar\} \endist
\end{istgame}

8.3 Fine-tuning positions: payoffs

You can change the position of payoffs with \texttt{TikZ} options: \texttt{xshift} and \texttt{yshift}. For example, you can do as shown in the following:

% Example: payoffs
\begin{istgame}
\istroot\{0\}
\istb\{Good\}\{1\}\{0,2\} \istb\{Bad\} \endist
\istroot\{1\}\{0-2\}\{<45\}\{\alpha\}\{\beta\}\{\langle xshift=-10pt \rangle \langle below \rangle \}
\istb\{\alpha\}\{al\}\{1,1\}\langle xshift=-10pt \rangle \langle below \rangle \}
\istb\{\beta\}\{ar\}\{-1,-1\}\{\langle yshift=-10pt \rangle \langle below \rangle \}
\endist
\end{istgame}

9 Growing direction of trees

You can draw a game tree that grows in any direction. By default, a game tree grows down and the child branches are arranged and named counterclockwise with respect to their parent node. When a tree grows down, the branches are arranged from left to right. When a tree grows to the right, the branches are arranged and named from bottom to top.

Sometimes you may want a tree with the branches arranged clockwise with respect to their parent node because it seems to look more natural, especially when a tree grows north or east. In \texttt{TikZ}, \texttt{grow'=<direction>} enables you to draw a tree developed clockwise.

To deal with the direction of the tree growth and the order of arranging branches, this package provides \texttt{\setistgrowdirection} as well as \texttt{\setistgrowdirection}.

% default: growing south counterclockwise
\def\xtgrow{grow}
\def\istdefault@grow\{south\} % tree growing direction
9.1 \texttt{\setistgrowdirection{}} – counterclockwise

Our first example is a tree drawn using the default values: growing \texttt{south} with branches going counterclockwise with respect to their parent nodes (from left to right).

\begin{istgame}
\setistOvalNodeStyle{.6cm}
\xtShowEndPoints[oval node]
\istrooto(0){0}+{12.5mm}..{3.45cm}+
\istb \istb \endist
\xtdistance{12.5mm}{11.5mm}
\istrooto(1)(0-1){1}
\istb{}{3}[center] \istb{}{4}[center]
\istb{}{5}[center] \endist
\istrooto(2)(0-2){2}
\istb{}{6}[center] \istb{}{7}[center] \endist
\end{istgame}

\begin{tikzpicture}
\node at (0,0) {0};
\node at (1,-1) {1};
\node at (2,-2) {2};
\node at (1,-3) {3};
\node at (1,-4) {4};
\node at (1,-5) {5};
\node at (2,-4) {6};
\node at (2,-5) {7};
\end{tikzpicture}

By default or with \texttt{\setistgrowdirection{south}}, the numbers written in the child nodes increase counterclockwise with respect to their parent nodes.

9.2 \texttt{\setistgrowdirection'} – clockwise

\begin{istgame}
\setistgrowdirection'{south}
\setistOvalNodeStyle{.6cm}
\xtShowEndPoints[oval node]
\istrooto(0){0}+{12.5mm}..{3.45cm}+
\istb \istb \endist
\xtdistance{12.5mm}{11.5mm}
\istrooto(1)(0-1){1}
\istb{}{3}[center] \istb{}{4}[center]
\istb{}{5}[center] \endist
\istrooto(2)(0-2){2}
\istb{}{6}[center] \istb{}{7}[center] \endist
\end{istgame}

\begin{tikzpicture}
\node at (0,0) {0};
\node at (-1,-1) {2};
\node at (1,-1) {1};
\node at (2,-2) {4};
\node at (1,-3) {5};
\node at (1,-4) {6};
\node at (-1,-4) {7};
\end{tikzpicture}

With \texttt{\setistgrowdirection'}{south}, the numbers written in the child nodes increase \texttt{clockwise} with respect to their parent nodes, which does not look natural.
9.3 Examples of rotating trees with \texttt{\setistgrowdirection}

This macro allows you to to rotate a game tree. When you rotate a game tree to the north or to the east, it is a good idea to use the swap version \texttt{\setistgrowdirection}’.

**Tips:** Though it is not necessary, it is suggested to use the following combinations of the macros and the directions.

- \texttt{\setistgrowdirection\{south\}}: branches going counterclockwise (from left to right)
- \texttt{\setistgrowdirection\{west\}}: branches going counterclockwise (downward)
- \texttt{\setistgrowdirection\{north\}}: branches going clockwise (from left to right)
- \texttt{\setistgrowdirection\{east\}}: branches going clockwise (downward)

**Remark:** When you use the swap version \texttt{\setistgrowdirection}’, using either \texttt{\istroot} or \texttt{\istroot'} gives you the same result.

### 9.3.1 A tree growing east – counterclockwise

\begin{verbatim}
\begin{istgame}
\setistgrowdirection{east}
% same codes as in Example 1
\setistOvalNodeStyle{.6cm}
\xtShowEndPoints[oval node]
\istrooto(0){0}+(12.5mm)..{3.45cm}+
\istb \istb \endist
\xtdistance{12.5mm}{11.5mm}
\istrooto(1){0-1}{1}
\istb\istb{}{3}[center] \istb\istb{}{4}[center]
\istb\istb{}{5}[center] \endist
\istrooto(2){0-2}{2}
\istb\istb{}{6}[center] \istb\istb{}{7}[center]
\endist
\end{istgame}
\end{verbatim}

Numbers increase upward (or counterclockwise).

### 9.3.2 A tree growing east – clockwise

\begin{verbatim}
\begin{istgame}
\setistgrowdirection\{east\}
% same codes as in Example 1
\setistOvalNodeStyle{.6cm}
\xtShowEndPoints[oval node]
\istrooto(0){0}+(12.5mm)..{3.45cm}+
\istb \istb \endist
\xtdistance{12.5mm}{11.5mm}
\istrooto(1){0-1}{1}
\istb\istb{}{3}[center] \istb\istb{}{4}[center]
\istb\istb{}{5}[center] \endist
\istrooto(2){0-2}{2}
\istb\istb{}{6}[center] \istb\istb{}{7}[center]
\endist
\end{istgame}
\end{verbatim}

Here, numbers increase downward (or clockwise). This looks more natural.

47
9.3.3 A tree growing north – counterclockwise

% Example 5: \setistgrowdirection{north}
\begin{istgame}
\setistgrowdirection{north}
% same codes as in Example 1
\setistOvalNodeStyle{.6cm}
\xtShowEndPoints[oval node]
\istrooto(0){0}+(12.5mm)..{3.45cm}+
\istb \istb \endist
\xtdistance{12.5mm}{11.5mm}
\istrooto(1){0-1}{1}
\istb{}{3}[center] \istb{}{4}[center]
\istb{}{5}[center] \endist
\istrooto(2){0-2}{2}
\istb{}{6}[center] \istb{}{7}[center]
\endist
\end{istgame}

Numbers increase from right to left (or counterclockwise).

9.3.4 A tree growing north – clockwise

% Example 6: \setistgrowdirection'\{north\}
\begin{istgame}
\setistgrowdirection'\{north\}
\setistOvalNodeStyle{.6cm}
\xtShowEndPoints[oval node]
\xtdistance{12.5mm}{11.5mm}
\istrooto(0){0}+(12.5mm)..{3.45cm}+
\istb \istb \endist
\xtdistance{12.5mm}{11.5mm}
\istrooto(1){0-1}{1}
\istb{}{3}[center] \istb{}{4}[center]
\istb{}{5}[center] \endist
\istrooto(2){0-2}{2}
\istb{}{6}[center] \istb{}{7}[center]
\endist
\end{istgame}

Numbers increase from left to right (or clockwise). This looks more natural.

9.3.5 \setxtgrowkey for one simple tree

\setxtgrowkey can be used to change the key between grow and grow’, which is useful especially for one simple tree, not a whole tree.

% \setxtgrowkey: definition
\NewDocumentCommand\setxtgrowkey{m}
{  \renewcommand\xtgrow{#1}
}
% #1 is either grow or grow’

The example below shows that the branches are arranged clockwise by \setistgrowdirection’. So you will need to use \setistgrow{grow} to locally get back to counterclockwise.
10 Information sets

10.1 \xtInfoset: standard version

The macro \xtInfoset draws an information set, connecting two decision nodes. The starred version \xtInfoset* prints the owner of an information set in input mode and text font style as set by \setistmathTF(*). For more details, see Section 10.2.3 on page 52.

Remark: This package provides macros to draw various types of information sets. They all include an option style [infostyle], which is equivalent to [semithick,densely dotted]. You can change the style or add more options to it by using the macro \setxtinfosetstyle. For more details, see Section 10.2.3 on page 52.
When you specify the owner of an information set, it appears (by default) above the midpoint of the two nodes. To change the location of the information set owner, you can use the abbreviations of directional words like a, l, ar, or bl. However, when you use other options with the position you cannot use the abbreviations. If you want an owner in red on the left, you can do it like [left, red] as shown below.

\begin{istgame}
\setistgrowdirection{east}
\setxtinfosetstyle{dashed} % changes line style
\istroot(0){1}+15mm..30mm+ \istb{a} \istb{b} \endist
\istroot(1)(0-1) \istb{c} \istb{d} \endist
\istroot(2)(0-2) \istb{c'} \istb{d'} \endist
\xtInfoset(1)(2){2}[left,red]
\end{istgame}

For a curved information set, you can do like, for example, \xtInfoset[bend left](1)(2). However this depends on the direction of tree growing or swapping the arrangement of branches. So, in order to draw a curved information set, it is recommended for you to use the curved version \xtCInfoset, documented in Section 10.3 on page 54.

10.2 \xtInfosetO: oval version

10.2.1 \xtInfosetO: basics

The oval version \xtInfosetO prints a bubble type (by default, a rounded rectangle) information set connecting two nodes, on the background layer by default. However, when you specify two identical nodes, it prints a densely dotted circle (by default) at the node to express a singleton information set.

Its starred version \xtInfosetO* prints the owner of an information set in input mode and text font style as set by \setistmathTF(*) (see page 42 for examples).

\% \xtInfosetO (from left to right)
\% syntax:
\% \xtInfosetO[bubble opt]{<from>}{<to>}{<owner>}{<owner opt>}{(min. height)}
\% defaults: connecting two nodes: \xtInfoset(coor1)(coor2)
\% [ -,draw,rectangle,samples,inner sep=Opt,transform shape,sloped,midway,
\% rounded corners=.5*<minimum height>*,\xtscale, minimum width=\n+<minimum height>,
\% minimum height=1em,inner sep=Opt,infoset style ]
\% (\<m>)(\<m>){\>()(\<m>)}{\}(){\]<em>}
\% option style for (all) information sets
\% infoset style=\{semithick,densely dotted\}

Here is an example that shows a bubble type information set as a rounded rectangle and a singleton information set as a circle.

\begin{istgame}
\istroot(0){1}+15mm..30mm+
\istb{a} \istb{b} \endist
\istroot(1)(0-1) \istb{c} \istb{d} \endist
\istroot(2)(0-2) \istb{c'} \istb{d'} \endist
\xtInfosetO(0)
\xtInfosetO(1)(2){2}
\end{istgame}
The height (or thickness) of an information set does not depend on scale, xscale, nor yscale.

\begin{istgame}[xscale=1.2,yscale=.7]
\istroot(0){1}+15mm..30mm+
\istb(a) \istb(b) \endist
\istroot(1)(0-1) \istb(c) \istb(d) \endist
\istroot(2)(0-2) \istb(c') \istb(d') \endist
\xtInfosetO(0)(0)
\xtInfosetO(1)(2){2}
\lerend{istgame}

The size of an information set adjusts accordingly to the scale values.

Remark:

- The (minimum) height of a bubble type information set is \texttt{1em} by default. This can be changed by the last optional argument of \texttt{xtInfosetO}, like \texttt{xtInfosetO(1)(2)(2em)}.
- The width of a bubble information set is \texttt{n1+minimum height} by default, where \texttt{n1} is the Euclidean distance (measured by \texttt{T\!i\!kZ}) between two nodes in an information set.

\begin{istgame}
\setistgrowdirection '{east}
\istroot(0){1}+15mm..30mm+
\istb(a) \istb(b) \endist
\istroot(1)(0-1) \istb(c) \istb(d) \endist
\istroot(2)(0-2) \istb(c') \istb(d') \endist
\xtInfosetO(0)(0)
\xtInfosetO(1)(2){2}
\lerend{istgame}

The shape of an information set does not depend on the direction of tree growing, either. You can change the shape to, for example, an ellipse by specifying it in the first bracket option list. You can also change the color of the bubble representing an information set by specifying it in the option list.

\begin{istgame}[yscale=1.5]
\setistgrowdirection '{south east}
\istroot(0) \istb \istb* \endist
\istroot(1)(0-2) \istb* \istb* \endist
\setxtinfosetstyle{fill=red!20,ellipse}
\xtInfosetO(0)(0-2){1}
\setxtinfosetstyle{solid,fill=blue!40,opacity=.5}
\xtInfosetO(1)(1-2){3}
\setxtinfosetstyle % restore defaults
\xtInfosetO(1)(1)
\lerend{istgame}

Note that, in the above example, \texttt{setxtinfosetstyle} is used to change the style of information sets. In order to restore the option value to default (i.e., \texttt{semithick,densely dotted}) just declare \texttt{setxtinfosetstyle}. (See Section 10.2.3 on page 52.)
10.2.2 Sloped information sets

With the istgame package, a sloped information set is not special. Just connect any two nodes using \xtInfoset or \xtInfosetO.

% Example: sloped infoset
\begin{istgame}
\istroot (0){Alice}+15mm..30mm+
\istb(A) \istb(B) \endist
\istroot (1)(0-1)<135>{Ben} \istb(C) \istb(D) \endist
\istroot (2)(0-2) \istb(a) \istb(b) \endist
\istroot (3)(1-2) \istb(a) \istb(b) \endist
\xtInfosetO(0)(0)(2em)
\xtInfosetO[rectangle](1)(1)
\xtInfosetO(3)(2){Cate}
\end{istgame}

If you want to have an owner sloped too, you need the option [sloped], as shown below.

% Example: sloped infoset with sloped text
\begin{istgame}
\setistgrowdirection '{east}
\istroot (0){Alice}+15mm..30mm+
\istb(A) \istb(B) \endist
\istroot (1)(0-1)<135>{Ben} \istb(C) \istb(D) \endist
\istroot (2)(0-2) \istb(a) \istb(b) \endist
\istroot (3)(1-2) \istb(a) \istb(b) \endist
\setistmathTF* 011<textbf>
\xtInfosetO(0)(0)
\xtInfosetO[rectangle,rounded corners=.2em](1)(1)
\xtInfosetO[ellipse,fill=blue!60](3)(2){Cate}[sloped,white](1.5em)
\end{istgame}

Warning: Issues in sloped labels in TikZ with asymmetric scales:

- The tree library in TikZ does not seem to treat sloped labels properly, when xscale or yscale is used asymmetrically.
- To cure this problem you can use \xtcureslopedlabelsNS for trees growing northwards and southwards and \xtcureslopedlabelsWS for trees growing eastwards and westwards. (See Section 6.2.3 and page 23, for more details with examples.)
- Note, however, that sloped labels for information owners printed by \xtInfosetO do not depend on scaling.

10.2.3 \setxtinfosetstyle

With the macro \setxtinfosetstyle you can change the style of all information sets, at once. For this end, a simple new style info set style is defined as follows:

% \setxtinfosetstyle
\NewDocumentCommand \setxtinfosetstyle { m }
{ \tikzset { infoset style/.style = { semithick , densely dotted , #1 } } }

You can change the line style of an information set or add more options to the option list like, for example, \setxtinfosetstyle{thin,dashed} or \setxtinfosetstyle{blue}.

If you want get the option values back to the default, then just declare \setxtinfosetstyle.
With \texttt{\setxtinfosetstyle}, you can also change the background color of information sets.

\begin{tikzpicture}[scale=0.8]
  \node (A) at (0,0) {Alice};
  \node (B) at (1,0) {Blue};

  \node (1) at (0.5,1) {a};
  \node (2) at (-1,1) {b};
  \node (3) at (1,1) {c};
  \node (4) at (2,1) {d};

  \draw[fill=blue!20] (1) circle (0.5cm);
  \draw[fill=red,ellipse,opacity=.2] (2) ellipse (0.5cm and 0.2cm);

  \draw (A) -- (1) -- (3);
  \draw (A) -- (2) -- (4);
\end{tikzpicture}

10.2.4 \texttt{\setxtinfosetlayer}

You can use the macro \texttt{\setxtinfosetlayer} to change the layer on which an information set lies from background (by default) to behind, main, above, or foreground, in that order. To go back to the default layer, just declare \texttt{\setxtinfosetlayer} or \texttt{\setxtinfosetlayer{}}.

\begin{tikzpicture}[scale=0.8]
  \node (A) at (0,0) {Alice};
  \node (B) at (1,0) {Blue};
  \node (1) at (0.5,1) {a};
  \node (2) at (-1,1) {b};

  \node (3) at (1,1) {c};
  \node (4) at (2,1) {d};
  \draw[fill=blue!20] (3) circle (0.5cm);
  \draw[fill=red,ellipse,opacity=.2] (2) ellipse (0.5cm and 0.2cm);

  \draw (A) -- (1) -- (3);
  \draw (A) -- (2) -- (4);
\end{tikzpicture}
10.3 \texttt{\textbackslash xtCInfoset}: curved version

With the macro \texttt{\textbackslash xtCInfoset} you can draw, by default, a \textit{curved} information set and even a \textit{skewed curved} information set, on the \textit{background layer} by default.

Its starred version \texttt{\textbackslash xtCInfoset*} prints the owner of an information set with the input mode and text font style as set by \texttt{\setistmathTF(*)} (see page 42 for examples).

10.3.1 Curved information sets with \texttt{\textbackslash xtCInfoset}: basics

The macro \texttt{\textbackslash xtCInfoset} connects two nodes with a curved information set like an arch that looks like a left-bent curve (by default) from start point to end point. The basic usage of \texttt{\textbackslash xtCInfoset} is the same as of \texttt{\textbackslash Infoset}.

Here is an example of drawing curved information sets.

10.3.2 Skewed \texttt{\textbackslash xtCInfoset}: full functions

Besides the basic functions, \texttt{\textbackslash xtCInfoset} has two additional optional arguments to control the shape of a curved information set. With the macro \texttt{\textbackslash xtCInfoset}, you can control the shape of an information set curve by using plot factor like \texttt{<1.3>} or \texttt{<0.7>} (by default \texttt{<1.3>}) and midpoint factor like \texttt{!.4!} or \texttt{!.6!} (by default \texttt{!.5!}).
By plot factor, we mean that it determines the maximum or minimum value of a curve. If the plot factor is greater than 1 it prints a concave curve, equal to 1 a straight line, and less than 1 a convex curve, connecting form left to right. For example, \texttt{\xtCInfoset(1)(2)<.7> (left to right)} and \texttt{\xtCInfoset(2)(1)<1.3> (right to left)} will give the same result.

\begin{verbatim}
\% \xtCInfoset: plot factor
\begin{istgame}[scale=1.5,font=\scriptsize]
\istroot(0){Ace}+15mm..30mm\istb\endist
\istroot(1)(0-1) \istb \istb \endist
\istroot(2)(0-2) \istb \istb \endist
\xtCInfoset*[dashed,blue](1)(2)<1.5>{1.5}
\xtCInfoset*[dashed,blue](1)(2)<1.3>{1.3}
\xtCInfoset*[dashed,blue](1)(2)<1>{1}
\setxtinfosetlayer{above}
\xtCInfoset*[dashed,blue](1)(2)<.7>{.7}
\setxtinfosetlayer{background}
\xtCInfoset*[solid,green,thick](2)(1)<1.3>
\xtCInfoset*[dashed,blue](1)(2)<.5>{.5}
\end{istgame}
\end{verbatim}

By midpoint factor, we mean that, roughly speaking, it determines the maximum or minimum point of a curve. If the midpoint factor is less than .5 the curve is positively skewed, and greater than .5 negatively skewed.

\begin{verbatim}
\% \xtCInfoset: midpoint factor (skewed)
\begin{istgame}[font=\scriptsize]
\istroot(0){Ace}+15mm..30mm\istb\endist
\istroot(1)(0-1) \istb \istb \endist
\istroot(2)(0-2) \istb \istb \endist
\xtdistance{10mm}{10mm}
\istroot(3)(1-1) \istb \istb \endist
\istroot(4)(1-2) \istb \istb \endist
\istroot(5)(2-1) \istb \istb \endist
\istroot(6)(2-2) \istb \istb \endist
\xtCInfoset*[dashed,blue](0)(1)<.7>{imperfect recall}[left]
\xtCInfoset*[dashed,blue](1-1)!-.05!(2-1)<1.6>{Ben}
\xtCInfoset*[dashed,blue](1-2)!+.9!(2-2)<1.4>{Cate}
\end{istgame}
\end{verbatim}

A curved information set drawn by \xtCInfoset does not depend on the tree growing direction. It does not depend on scaling nor swapping branches, either.

\begin{verbatim}
\begin{istgame}[scale=.85,font=\scriptsize]
\setistgrowdirection'{east}
\istroot(0){Ace}+15mm..30mm\istb\endist
\istroot(1)(0-1) \istb \istb \endist
\istroot(2)(0-2) \istb \istb \endist
\xtdistance{10mm}{10mm}
\istroot(3)(1-1) \istb \istb \endist
\istroot(4)(1-2) \istb \istb \endist
\istroot(5)(2-1) \istb \istb \endist
\istroot(6)(2-2) \istb \istb \endist
\xtCInfoset*[dashed,blue](0)(1)<.7>{imperfect recall}[left]
\xtCInfoset*[dashed,blue](1-1)!-.05!(2-1)<1.6>{Ben}
\xtCInfoset*[dashed,blue](1-2)!+.9!(2-2)<1.4>{Cate}
\end{istgame}
\end{verbatim}

55
10.4 \texttt{\textbackslash xtCInfoset}: curved oval version

The macro \texttt{\textbackslash xtCInfoset} enables you to draw a \emph{curved bubble type} information set and even a \emph{skewed curved bubble type} information set on the \emph{background layer} by default.

Its starred version \texttt{\textbackslash xtCInfoset}\texttt{\textbackslash asterisk} prints the owner of an information set in input mode and text font style in accordance with \texttt{\textbackslash setistmathTF\textbackslash asterisk} (see page 42 for examples).

10.4.1 Curved bubble type information sets with \texttt{\textbackslash xtCInfoset}: basics

The basic usage of \texttt{\textbackslash xtCInfoset} is the same as of \texttt{\textbackslash xtInfoset}. If the two mandatory coordinates are \emph{identical}, a \emph{circle} (but not a rectangle) is drawn to represent a \emph{singleton information set}, like the case of \texttt{\textbackslash xtInfoset}.

The shape of information sets does not depend on the direction of tree growing. You also can change the background color and the height, like \texttt{(1.5em)} as the last option.
10.4.2 Skewed $\texttt{xtCInfosetO}$: full function

Besides the basic functions, the macro $\texttt{xtCInfosetO}$ has two additional optional arguments to control the shape of a bubble representing a curved information set. The two optional arguments are plot factor and midpoint factor. (The meanings are documented on page 54.)

The plot factor makes a curved information set higher or lower and is used in angle brackets, like $<1.5>$ or $<.7>$ (by default $<1.3>$), right after the two mandatory coordinates. The midpoint factor controls skewness of a curved information set and is used in between the two mandatory arguments, like $!.35!$ (by default $!.5!$).

\begin{verbatim}
% $\texttt{xtCInfosetO}$ : full definition
% syntax: $\texttt{xtCInfosetO}[^{bubble opt}][^{<from}>][midpoint factor]^{^{<to>}}$
% defaults: connecting two nodes: $\texttt{xtInfoset(coor1)(coor2)}$
% [-,draw,samples,inner sep=0pt, minimum size=1em, infoset style]
% infoset style = { semithick , densely dotted }

% $\texttt{xtCInfosetO}$: skewed
\begin{istgame}[font=\scriptsize]
\istroot (0){Ace}+15mm..30mm+\istb \endist
\istroot (1)(0-1) \istb \endist
\istroot (2)(0-2) \istb \endist
\xtdistance{10mm}{10mm}
\istroot (3)(1-1) \istb \endist
\istroot (4)(1-2) \istb \endist
\istroot (5)(2-1) \istb \endist
\istroot (6)(2-2) \istb \endist
\xtCInfosetO[fill=blue!20] (1)(0){imperfect recall}[left]
\xtCInfosetO[dashed,blue](1)(2)<.7>{Blue}
\xtCInfosetO(1-1)!1.35!(2-1){Ben}
\xtCInfosetO[fill=red!40,opacity=.5] (1-2)!1.65!(2-2)<1.5>{Cate}
\end{istgame}
\end{verbatim}

Remark:

- When using $\texttt{xtCInfosetO}$, the recommended range of the midpoint factor is between $.4$ and $.6$, at most between $.35$ and $.65$, otherwise you might get a result with which you are not satisfied. (You can see below Section 10.5.1 on this issue, only when you are interested in.)

- You do not need to bother if the tree is swapped, because the package internally takes care of that instead.

- An information set drawn by $\texttt{xtCInfosetO}$ is appropriately adjusted with $\texttt{scale}$, $\texttt{xscale}$, or $\texttt{yscale}$, but the height (or thickness) is independent of the scale values. (You can see below Section 10.5.2 on the issue of scaling, only when you are interested in.)

10.5 Fine-tuning $\texttt{xtCInfosetO}$: Not for most users

10.5.1 $\texttt{xtCInfosetOTurnX}$: too high or too low midpoint factor

What if you really want to use the midpoint factor close to 0 or 1? In this case, you will possibly get some unsatisfactory result as shown in the example below. (Note that this being unsatisfied might not happen if you use the midpoint factor within the recommended range.)
The package provides the macro \texttt{xtCInfosetOTurnX} to make it straight.

First, find the blue circle and the red circle in the above example. Let us call by \texttt{X} the (blue) circle at the beginning of an information set and by \texttt{Y} at the end. With \texttt{xtCInfosetOTurnX} you can turn these circles to get a better result.

In the following example, \texttt{xtCInfosetOTurnX{45}{0}} and \texttt{xtCInfosetOTurnX{0}{-45}} are used to correct the result. If you omit the second angle like, for example, \texttt{xtCInfosetOTurnX{45}}, it is equivalent to \texttt{xtCInfosetOTurnX{45}{-45}}, meaning that it turns the both circles symmetrically.

Remark:

- \texttt{xtCInfosetOTurnX} works just once only for the following \texttt{xtCInfosetO}, without affecting any other.
- You can also change the layer of an information set, as shown in the previous example, by using \texttt{setxtinfosetlayer} (see Section 10.2.4).
10.5.2 Scaling information sets according to the value of Ti\textit{kZ} scale

The height (or thickness) of bubble type information sets drawn by \texttt{\textbackslash xtInfosetO} and \texttt{\textbackslash xtCInfosetO}, does not depend on the values of Ti\textit{kZ} scales. What if you want to make them scaled according the value of Ti\textit{kZ} symmetric scale, for example, \texttt{[scale=.5]}? Then use, instead, \texttt{[scale=.5,xscale=1]} or equivalently \texttt{[yscale=1,scale=.5]} in any order of the options.

Here is an ordinary example of using \texttt{[scale=.5]}.

\begin{istgame}[scale=.5]
\xtdistance{40mm}{30mm}
\istroot(0)+40mm..40mm+
\istb \istb \istb \endist
\istroot(1)(0-1) \istb \istb \istb \endist
\istroot(2)(0-2) \istb \istb \istb \endist
\istroot(3)(0-3) \istb \istb \istb \endist
\xtCInfosetO(0)(0)(2em)
\xtInfosetO[fill=green](1)(2)[green]
\xtCInfosetO(1)(3){player}(1.5em)
\end{istgame}

Here is an example to use \texttt{[scale=.5,xscale=1]}, where the height (or thickness) of information sets is scaled according to the value of Ti\textit{kZ} scale.

\begin{istgame}[scale=.5,xscale=1]
\xtdistance{40mm}{30mm}
\istroot(0)+40mm..40mm+
\istb \istb \istb \endist
\istroot(1)(0-1) \istb \istb \istb \endist
\istroot(2)(0-2) \istb \istb \istb \endist
\istroot(3)(0-3) \istb \istb \istb \endist
\xtCInfosetO(0)(0)(2em)
\xtInfosetO[fill=green](1)(2)
\xtCInfosetO(1)(3){player}(1.5em)
\end{istgame}

Remark: (Not for most users) How the \texttt{istgame} environment works with the Ti\textit{kZ} scales, regarding oval type information sets.

- In the \texttt{istgame} package, it is intended that the height (or thickness) of any bubble type information set is not affected by the values of Ti\textit{kZ} scale, unless users change it. (This is a desirable feature.)
- The shapes of bubble type information sets work perfectly fine, as intended, when:
  - scale only (not with xscale nor yscale) is used in the environment option list
  - and either xscale or yscale or both (not with scale) are used in the option list.
- Now, this is how the \texttt{istgame} environment works when scale is used with xscale or yscale.
  - When this is the case, the value of scale is not taken care of by the package. Only the values of xscale and yscale are internally extracted and used to compensate the distortion of the shapes of information sets caused by the asymmetric scaling.
  - For example, \texttt{[scale=.5]} and \texttt{[scale=.5,xscale=1]} are treated as different (with respect to bubble type information sets) by the \texttt{istgame} environment, while the \texttt{tikzpicture} environment treats them as equal. With the latter, the \texttt{istgame} environment does not extract the value of scale to use it internally, but does only the value of xscale. So scale can affect the sizes of bubble type information sets.
  - Still, the shapes of bubble type information sets drawn by this package are not distorted by the mixed (even asymmetrical) use of Ti\textit{kZ} scales. (In the case of asymmetric scaling,
there is one exception for \texttt{\textbackslash cntmAInfosetO}, which is equipped with an option to cure the distortion. See Section 11.5.3 on page 74 for more details.)

\section{Continuum of branches}

The package \texttt{istgame} provides the macro \texttt{\textbackslash istrootcntm} and its several variants (all prefixed by \texttt{\textbackslash istroot} or \texttt{\textbackslash istrooto}) to express a continuum of branches and an action taken. As you will see, all the supplemental macros to a continuum of branches are prefixed by \texttt{\textbackslash cntm}.

Two types of graphic objects, a triangle type and an arc type, are provided in the package to represent a continuum of branches. So you can start by choosing one type of a continuum of branches.

\subsection{\texttt{\textbackslash istrootcntm: standard continuum version}}

\subsection*{\texttt{\textbackslash istrootcntm: basics}}

The standard version \texttt{\textbackslash istrootcntm} prints a background triangle, in black!25 by default, representing a continuum of branches, on the background layer by default. It works just like \texttt{\textbackslash istroot}, for one exception: it controls, by the last two options \texttt{+\textbackslash cntmlevdist\ldots \textbackslash cntmsibdist+}, the distances for only the background triangle, but not the \texttt{\textbackslash istb}'s following it. You can regard \texttt{\textbackslash istrootcntm} as the sum \texttt{\textbackslash istroot + \textbackslash cntn}.

% \texttt{\textbackslash istrootcntm}
% syntax:
\texttt{\textbackslash istrootcntm\[<grow keyval>,<opt>\](<coor1>)(<coor2>)[<node style>,<opt>]
% <owner label angle>{<owner>}+<cntm-levdist>..<cntm-sibdist>+
% defaults:
% [south]<(m>)(0,0)[decision node]<above>{}\{+8mm..24mm+
% arguments: (coor1) is mandatory, all others are optional arguments
% [grow] % the direction of growing <default: south>
% (coor1) % name of the (sub)root: mandatory
% (coor2) % the (sub)root is at (coor2) <default: (0,0)>
% [node style] % node style <default: decision node>
% <angle> % position of owner name <default: above>
% {owner} % name of the owner of the (sub)root
% +cntmlevdist..cntmsibdist+ % <defaults: 8mm,24mm>

The background is an isosceles triangle, with the height of \texttt{\textbackslash cntmlevdist} (8mm by default) and the base length of \texttt{\textbackslash cntmsibdist} (24mm by default).

% \texttt{\textbackslash istrootcntm (\textbackslash istroot + \textbackslash cntn)}
% \begin\{istgame\}[scale=1.2]
% \cntmdistance{8mm}
% \istrootcntm(0){I}
% \istb{x}[r] \istbm \endist
% \xtdistance{10mm}{20mm}
% \istroot(1)(0-1){[label distance=-3pt]120}{II}
% \istb{Y}[al]{x,1-x}
% \istb{N}[ar]{0,0}
% \endist
% \end\{istgame\}

The action taken by the owner of the root has the sibling distance (to a missing branch represented by \texttt{\textbackslash istbm}, if exists) of one third (by default) of the base length of the continuum triangle.
You can change the level and sibling distances of an individual continuum triangle by specifying the last two options of the macro \texttt{\textbackslash istrootcntm} like, for example, +10mm..15mm+. Note that these options do not affect the action sibling distance.

Remark: Below we will see the convenient distance changers \texttt{\textbackslash cntmdistance} and \texttt{\textbackslash cntmdistance*} that can be used to change the distances for all types of continua. These macros also have a control on the action sibling distance.

\begin{itemize}
\item Once the work of \texttt{\textbackslash istrootcntm} is completed by \texttt{\textbackslash endist}, \texttt{\textbackslash istrootcntm} internally produces three node coordinates: (cntm), (cntm-1), and (cntm-2).
\item You can use these coordinates after \texttt{\textbackslash istrootcntm} and before another \texttt{\textbackslash istrootcntm} or one of its variant, like \texttt{\textbackslash istrootcntmA}, is used.
\end{itemize}
\texttt{\textbackslash istrootcntm'}: swap version

The swap version \texttt{\textbackslash istrootcntm'} arranges branches clockwise just like \texttt{\textbackslash istroot'} does. With \texttt{\textbackslash setistgrowdirection'}, both \texttt{\textbackslash istrootcntm} and its swap version \texttt{\textbackslash istrootcntm'} end up with the same result.

\begin{verbatim}
% \texttt{\textbackslash istrootcntm'} (swap version)
\begin{istgame}[scale=1.2]
\setistgrowdirection'{east}
\istrootcntm'(0){I}
\istbm {x}[b] \isted \endist
\xtdistance{10mm}{20mm}
\istroot(1)(0-1)<[\text{label distance}=-3pt]120>{II}
\istb{Y}[a]{x,1{-}x} \isted
\istb{N}[b]{0,0} \isted
\end{istgame}
\end{verbatim}

11.1.2 \texttt{\textbackslash cntmdistance}

\texttt{\textbackslash cntmdistance} working for \texttt{\textbackslash istrootcntm} (and \texttt{\textbackslash istrootcntmA}) is analogous to \texttt{\textbackslash xtdistance} working for \texttt{\textbackslash istroot}. \texttt{\textbackslash cntmdistance} controls the height and width of a background triangle representing a continuum, but it has one more function than \texttt{\textbackslash xtdistance}. With \texttt{\textbackslash cntmdistance} you can also control the sibling distance of action branches (\textit{action sibling distance}) that are taken by the owner of the root.

\begin{verbatim}
% \texttt{\textbackslash cntmdistance}
% syntax:
% \texttt{\textbackslash cntmdistance}{<cntm levdist>}{<cntm sibdist>}{<action sibdist>}
% defaults: <cntm levdist> is mandatory, and all others are optional
% {8mm}{3*<cntm lev dist>}{(1/3)*<cntm sib dist>}
\end{verbatim}

Note that the first argument of \texttt{\textbackslash cntmdistance} is \textit{mandatory}, and all the others are optional. For example, \texttt{\textbackslash cntmdistance}{10mm}{20mm} means the followings:

\texttt{\textbackslash cntmlevdist} = 10mm, \texttt{\textbackslash cntmsibdist} = 20mm, and \texttt{\textbackslash cntmactsibdist} = (1/3)*\texttt{\textbackslash cntmsibdist}, from now on.

And \texttt{\textbackslash cntmdistance}{10mm}{20mm}{3mm} means now \texttt{\textbackslash cntmactsibdist} = 3mm.

\begin{verbatim}
% \texttt{\textbackslash cntmdistance}
\begin{istgame}[font=\scriptsize]
\xtdistance{10mm}{20mm}
\cntmdistance{10mm}{20mm}{3mm}
\istrootcntm(1){(1)} \textit{peroid 1}
\istb{x}[r] \isted
\istroot(A1)(1-1)<[\text{label distance}=-3pt]135>{(2)}
\istb{A}[a]{x,1{-}x} \isted\istb{R}[ar] \isted
\istrootcntm(2)(A1-2){(2)} \textit{peroid 2}
\istb{y}[r] \isted
\istroot(A2)(2-1)<[\text{label distance}=-3pt]135>{(1)}
\istb{A}[a]{1{-}y,y} \isted\istb{R}[ar] \isted
\istrootcntm(3)(A2-2){(1)} \textit{peroid 3}
\istb{x}[r] \isted
\istroot(A1)(3-1)<[\text{label distance}=-3pt]135>{(2)}
\istb{A}[a]{x,1{-}x} \isted\istb{R}[ar]{0,0} \isted
\end{istgame}
\end{verbatim}
11.1.3 \texttt{\textbackslash cntmdistance*}: combined with \texttt{\textbackslash xtdistance}

The macro \texttt{\textbackslash cntmdistance*} incorporates \texttt{\textbackslash cntmdistance} with \texttt{\textbackslash xtdistance}.

\begin{verbatim}
% \texttt{\textbackslash cntmdistance*}
% syntax:
% \texttt{\textbackslash cntmdistance*}{\texttt{\textbackslash lev dist}}{\texttt{\textbackslash sib dist}}{\texttt{\textbackslash action sib dist}}
% defaults:
% \{15mm\}{15mm}{(1/3)*\texttt{\textbackslash sib dist}}
\end{verbatim}

The macro \texttt{\textbackslash cntmdistance} does not have anything to do with \texttt{\textbackslash xtdistance}, but the starred version \texttt{\textbackslash cntmdistance*} does. For example, \texttt{\textbackslash cntmdistance*{10mm}{20mm}} means the followings:

\texttt{\textbackslash cntmlevdist = \textbackslash xtlevdist = 10mm, \texttt{\textbackslash cntmsibdist = \textbackslash xtsibdist = 20mm, and}}
\texttt{\textbackslash cntmactsibdist = (1/3)*\texttt{\textbackslash sib dist}}, from now on.

And \texttt{\textbackslash cntmdistance*{10mm}{20mm}{3mm}} means now \texttt{\textbackslash cntmactsibdist = 3mm}.

11.1.4 \texttt{\textbackslash cntmpreset}: controlling continuum triangles

The macro \texttt{\textbackslash cntmpreset} controls the line style and color, size (via the shrink factor with 1 as default), and fill color of a continuum triangle drawn by \texttt{\textbackslash istrootcntm}.

\begin{verbatim}
% \texttt{\textbackslash cntmpreset}
% syntax:
% \texttt{\textbackslash cntmpreset}{\texttt{\textbackslash line style}}{\texttt{\textbackslash opts}}{\texttt{\textbackslash shrink factor}}{\texttt{\textbackslash fill color}}
% defaults:
% \{-,solid,draw=black!25,fill=<draw color>\}{1}\[
\end{verbatim}

To change the background color you can do like, for example, either \texttt{\textbackslash cntmpreset[blue!20]} or \texttt{\textbackslash cntmpreset{1}[blue!20]}. However, like \texttt{\textbackslash cntmpreset[blue!20][red!20]}, if you specify both of the bracket optional arguments, then the first one wins. The second bracket option is just for your convenience to put fill color easily.
You can also draw a smaller triangle, by specifying shrink factor as a decimal number in the curly braces.

\begin{istgame}[font=\scriptsize] 
\cntmdistance*(10mm)(20mm) 
\cntmpreset{.7} 
\istrootcntm(1) \istb \endist 
\istroot(A1)(1-1) \istb \istb \endist 
\cntmpreset[draw=blue,dashed]{.8}[red!20] 
\istrootcntm(2)(A1-2) \istb \istbm \endist 
\istroot(A2)(2-1) \istb \istb \endist 
\end{istgame}

11.1.5 \cntmpreset*: simple triangles with no background color

The macro \cntmpreset* works like \cntmpreset, for one exception: it prints, by default, a simple triangle with sides drawn, but with no background color.

\begin{doccode} 
% \cntmpreset* 
% syntax: \cntmpreset*[<line style>,<opts>]{<shrink factor>}[<fill color>] 
% defaults 
[-,solid,draw=black,fill=none]{1}[] 
\end{doccode}

Remark: In order to restore the standard triangle options (i.e., [-,draw=none,black!25]), you can just declare \cntmpreset followed by nothing. More precisely, if you use \cntmpreset without specifying the first bracket option, like \cntmpreset{.7}, the default options are restored.

\begin{doccode} 
% \cntmpreset* 
\begin{istgame}[font=\scriptsize] 
\cntmdistance*(10mm)(20mm) 
\cntmpreset* 
\istrootcntm(1) \istb \endist 
\istroot(A1)(1-1) \istb \istb \endist 
\cntmpreset*[fill=blue!20] 
\istrootcntm(2)(A1-2) \istb \istbm \endist 
\istroot(A2)(2-1) \istb \istb \endist 
\cntmpreset % restore default triangle options 
\istrootcntm(3)(A2-2) \istb \istbm \endist 
\istroot(A1)(3-1) \istb \istb \endist 
\end{istgame}

You can also draw a smaller triangle, by specifying shrink factor as a decimal number in the curly braces.
11.2 \texttt{\textbackslash istrootcntmA}: continuum arc version

11.2.1 \texttt{\textbackslash istrootcntmA}: basics

The macro \texttt{\textbackslash istrootcntmA} works just like \texttt{\textbackslash istroot}, but it draws an arc to express a continuum of branches. Here \texttt{cntmA} is an abbreviation of \texttt{cntmarc}. You can regard \texttt{\textbackslash istrootcntmA} as the sum \texttt{\textbackslash istroot} + \texttt{cntmA}.

Remark:

- There is one important difference between \texttt{\textbackslash istroot} and \texttt{\textbackslash istrootcntmA}. By the last two options (for example, +10mm..20mm+), the \texttt{\textbackslash istrootcntmA} controls the level and sibling distances of a continuum arc, while \texttt{\textbackslash istroot} controls the level and sibling distances of action branches (represented by the \texttt{\textbackslash istb}) following it.

- Note that the distance changers \texttt{\textbackslash cntmdistance} and \texttt{\textbackslash cntmdistance*} work the same for \texttt{\textbackslash istrootcntmA} as well as \texttt{\textbackslash istrootcntm}. \textit{(There is no such thing as \textbackslash cntmAdistance.)}

The default level and sibling distance of a continuum of branches are 8mm and 3*8mm. With the default distances, the distance between the root and the lowest point of the arc is one third longer (with all default values) than the continuum level distance.
With the macro \texttt{\textbackslash istbA} you can draw a single branch ending up with an endpoint on the continuum arc. Just remember \texttt{\textbackslash istb} always reaches the red dashed line and \texttt{\textbackslash istbA} the blue dashed line. For more details, see below Section 11.2.2 (though it is not for first readers).

\istrootcntmA'  

The swap version \texttt{\istrootcntmA'} arranges branches clockwise, like \texttt{\istroot'} does. Note also that if you use \texttt{\setistgrowdirection'}, then using the swap version makes no difference in results.

11.2.2 \istrootcntmA and \istbA

As discussed in Section 5.5 on page 29 \texttt{\textbackslash istb} and \texttt{\textbackslash istbA} can be used interchangeably (but with one exception). However, when it comes to \texttt{\istrootcntmA}, \texttt{\textbackslash istbA} has a unique function: it reaches a continuum arc.
To be precise, \texttt{\textbf{\textit{istbA}}} reaches the \textit{blue dashed line}, as shown in the above picture, while \texttt{\textbf{\textit{istb}}} reaches the \textit{red dashed line}. So when \texttt{\textbf{\textit{istbA}}} is the only child of the root, it arrives exactly at the continuum arc.

Moreover, as also discussed in Section 5.5 on page 29, \texttt{\textbf{\textit{istbA}}} has one more function than \texttt{\textbf{\textit{istb}}}: it can easily change its level distance by using the first parenthesis option. This allows you to make any \texttt{\textbf{\textit{istbA}}} reach the continuum arc, but after some trial and errors.

\subsection*{11.2.3 \texttt{\textbf{\textit{cntmA}}}: controlling continuum arcs and wedges}

With \texttt{\textbf{\textit{cntmA}}}, you can control the line style, position (via \texttt{arc position factor}), and curvature (via \texttt{arc plot factor}) of a continuum arc. You can also change the background color (by default transparent) of the \textit{wedge} formed by a continuum arc.

You can change the line style or the color of a continuum arc, in the usual Ti\textit{k}Z way, like \texttt{[dashed,thick,blue]}. If you do not specify any optional argument, the corresponding argument will have the default values. So, by declaring \texttt{\textbf{\textit{cntmA}}} followed by nothing, you can restore all the default values.
Remark:

- A continuum arc connects the two endpoints, by default, of the two outermost branches. If the arc position factor (between zero and one) is less than {0.5} the arc gets closer to the root, and if greater, then it gets closer to the endpoints.

- What if the factor is greater than 1? You try!

\% \cntmApreset (with \istrootcntmA and \istbA)
\begin{istgame}
\cntmdistance*{10mm}{20mm}
\cntmApreset[dashed,blue,thick]
\istrootcntmA(1)
  \istbA \endist
\istroot(A1)(1-1)
  \istb \istb \endist
\cntmApreset[dashed]{.6}
\istrootcntmA(2)(A1-2)
  \istb \endist
\istroot(A2)(2-1)
  \istb \istb \endist
\cntmApreset % restore defaults
\istrootcntmA(3)(A2-2)
  \istbA \endist
\istroot(A1)(3-1)
  \istb \istb \endist
\end{istgame}

Note that \istbA (if it is the only child) always prints a branch with its endpoint exactly on the arc. Note also that the effect of \cntmApreset goes on until another \cntmApreset, which restores the values to defaults, is used.

Remark:

- As for the curvature, if the factor (usually, greater than 1) typed in angle brackets is smaller than the default value <1.333> the arc gets flatter, and if larger it gets sharper.

- If the factor is 1, it becomes a straight line. What if the factor is less than 1? You guess!

\% \cntmApreset<plot factor>
\begin{istgame}
\cntmdistance*{10mm}{20mm}
\cntmApreset<1.2>
\istrootcntmA(1)
  \istbA \endist
\istrootcntmA(2)(1-1)
  \istbA \endist
\cntmApreset[tension=1.12]<1.508>
\istrootcntmA(2)(1-1)
  \istbA \endist
\istrootcntmA(3)(2-1)
  \istbA* \endist
\end{istgame}

In order to change the background color of the wedge formed by an arc, you should use the second bracket option. The first one is for a continuum arc line.
\cntmAlayerpreset: continuum arc layers

The color filled wedge lies on the behind layer, by default. You can change the layer by using \cntmAlayerpreset. Available layers are: background, behind, main, above, and foreground, in that order. To restore the default layer (i.e., behind), just declare \cntmAlayerpreset.

\begin{istgame}
\cntmApreset{.5}[blue!20]
\cntmAlayerpreset{background}
\istrootcntmA(1) \istbA \endist
\istroot(A1)(1-1) \istb \istb \endist
\istrootcntmA(2)(1-2) \istbA \endist
\istroot(A2)(2-1) \istb \istb \endist
\cntmAlayerpreset[dashed,blue,thick][red!20]
\cntmAlayerpreset \% restore default (behind)
\istrootcntmA(3)(2-2) \istbA \endist
\istroot(A3)(3-1) \istb \istb \endist
\xtInfosetO[fill=green](A1)(A3)
\end{istgame}

11.3 \istrootocntm and \istrootocntmA: oval continuum versions

In some cases, you may want to use \istrootocntmA, the oval version of \istrootcntmA. It can be regarded as the sum \istrooto + cntmA. The swap version \istrootocntmA' is also provided.

\begin{istgame}
\setistEllipseNodeStyle[white]
\cntmAlayerpreset[dashed]
\istrootocntmA(1){A} \istbA{x}[r] \istbm \endist
\istrooto(1a)(1-1){B}
\istb(Y)[al] \istb(N)[ar] \endist
\istrootocntmA(2)(1a-2){B} \istbA{y}[x] \istbm \endist
\istrooto(2a)(2-1){A}
\istb(Y)[al] \istb(N)[ar] \endist
\end{istgame}

It is anyway possible to use \istrootocntm, which is the oval version of \istrootcntm. It can be regarded as the sum \istrooto + cntm. You can also use its swap version \istrootocntm'.

\begin{istgame}
\cntmAlayerpreset[dashed]
\istrootcntm(1a)(1-1){B}
\istb(Y)[al] \istb(N)[ar] \endist
\istrooto(1a-2)(B) \istbA{y}[x] \istbm \endist
\istrooto(2a)(2-1){A}
\istb(Y)[al] \istb(N)[ar] \endist
\end{istgame}
11.4 Doing some chores: not for most users

11.4.1 \cntmistb: controlling outermost branches

The macro \cntmistb controls the outermost branches of a continuum triangle. This works similar to \istb, but you need to use this macro right before \istrootcntm. And it works just once only for the immediately following \istrootcntm and its close variants, but not for the continuum arc versions.

Remark: The macro \cntmistb* prints solid nodes at the endpoints of the outermost branches.

- The effect of printing solid nodes affects all the following continua. It can be turned off by the starred version \xtHideEndPoints* (see Section 11.4.3, for more details).

\begin{verbatim}
% syntax: \cntmistb
\cntmistb[<branch opt>]{<left action label>}[<pos>]{<right action label>}[<pos>]
% defaults: all arguments are optional
[,-,draw=none]{}[]{}
% \cntmistb (cntm outermost branches)
\begin{istgame}[font=\scriptsize]
\cntmxdistance*{10mm}{20mm}{3mm}
\cntmistb[draw=blue,dashed]{x=0}{1}[x=1]{above,sloped}
\istrootcntm(1)
  \istb \istbm \endist
\istroot(A1)(1-1)
  \istb \istbm \endist
\cntmistb*[blue]{x=0}{1}[x=1]{above,sloped}
\istrootcntm(2)(A1-2)
  \istb \istbm \endist
\istroot(A2)(2-1)
  \istb \istbm \endist
\setistmathTF*000{texttt}
\cntmistb[red,draw=blue,thick]{x=0}{1}[x=1]{above,sloped}
\istrootcntm(3)(A2-2)
  \istb \istbm \endist
\istroot(A1)(3-1) \istb \istbm \endist
\end{istgame}
\end{verbatim}

\cntmistb(*) prints action labels in math mode by default, and the input mode can be changed by \setistmathTF(*).
11.4.2 \texttt{cntmAistb}: controlling outermost branches

The macro \texttt{cntmAistb} controls the outermost branches of a continuum arc. This works similar to \texttt{istb}, but you need to use this macro right before the \texttt{istrootcntmA}. And it works just once for the immediately following \texttt{istrootcntmA} and its close variants, but not for the continuum triangle versions.

The macro \texttt{cntmAistb*} prints solid nodes at the endpoints of the outermost branches. The effect of printing solid nodes affects all the following continua. It can be turned off by the starred version \texttt{xtHideEndPoints*} (see Section 11.4.3, for more details).

\begin{verbatim}
% syntax: \cntmAistb\[<branch opt>\][{<left action label>}]\[<pos>\][{<right action label>}]\[<pos>\]
% defaults: all arguments are optional
[-]{}[]{}
\end{verbatim}

Note that \texttt{cntmAistb} controls outermost branches of a continuum, while \texttt{cntmApreset} controls the arc line.

\begin{verbatim}
\begin{istgame}[font=\scriptsize]
cntmdistance*{10mm}{20mm}
cntmAistb[dotted]{x=0}{above,sloped}{x=1}{above,sloped}
cntmApreset[dashed,blue,thick]{.7}
\istrootcntmA(1) \istbA \endist
\istroot(A1)(1-1) \istbA \istbA \endist
\end{istgame}
\end{verbatim}

\texttt{cntmAistb(*)} prints action labels in math mode by default, and the input mode can be changed by \texttt{setistmathTF(*)}.

\begin{verbatim}
\begin{istgame}[font=\scriptsize]
cntmdistance*{10mm}{20mm}
cntmAistb[dashed]{x=0}{l}{x=1}{above,sloped}
cntmApreset[dashed]{.7}
\istrootcntmA(1) \istbA \endist
\istroot(A1)(1-1) \istbA \istbA \endist
\istroot(A2)(2-1) \istbA \istbA \endist
\setistmathTF*0{\texttt{t}}
cntmAistb[draw=blue,thick]{x=0}{l}{x=1}{above,sloped}
cntmApreset{.9}{blue!20}
\istrootcntmA(3)(A2-2) \istbA \istbA \endist
\istroot(A2)(3-1) \istbA \istbA \endist
\end{istgame}
\end{verbatim}

It is not impossible to print two endpoints of a continuum arc, even thought it does not make any game theoretical sense. You can easily do that by using \texttt{cntmAexpostShowEndPoints} right after \texttt{istrootcntmA}.

\begin{verbatim}
\begin{istgame}[font=\scriptsize]
cntmAistb*
cntmApreset{.7}
\istrootcntmA(1) \istbA \endist
cntmAexpostShowEndPoints
\istrootcntmA(A1)(1-1) \istbA* \endist
\cntmAexpostShowEndPoints[oval node][box node]
\end{istgame}
\end{verbatim}
\section*{11.4.3 \texttt{xtShowEndPoints}}

The starred version \texttt{xtShowEndPoints} has additional control over the endpoints of outermost branches of a continuum, when it is used with \texttt{cntmistb} or \texttt{cntmAistb}. However, the starred versions \texttt{cntmistb*} and \texttt{cntmAistb*} do not obey \texttt{xtShowEndPoints}. They anyway print solid nodes at the endpoints of outermost branches and the effect goes on until it is blocked by \texttt{xtHideEndPoints} or \texttt{xtHidePoints*}.

\texttt{xtShowEndPoints*} competes with each of the starred versions \texttt{cntmistb*} and \texttt{cntmAistb*} for the control over \texttt{cntmistb} and \texttt{cntmAistb} with respect to the endpoints of outermost branches. Whichever comes later wins.

The starred version \texttt{xtHideEndPoints*} works only for the endpoints of the outermost branches to turn off the effects of \texttt{xtShowEndPoints*}, \texttt{cntmistb*}, and \texttt{cntmAistb*}. But whichever comes later wins.

\textbf{Remark}: Note that the non-starred versions \texttt{xtShowEndPoints} and \texttt{xtHideEndPoints} have nothing to do with the outermost endpoints of continua.

In the example below, \texttt{cntmistb*} overrides the effect of \texttt{xtShowEndPoints*[oval node]}, with respect to the outermost endpoints of continua.

\begin{lstlisting}[language=istgame]
\begin{istgame}
\cntdistance*{8mm}{16mm}
\cnpreset*
\xtShowEndPoints*[oval node]
\cntmistb
\istrootcntm(1)  \istb  \endist
\istroot(A1)(1-1) \istb \istb  \endist
\cntmistb
\istrootcntm(2)(A1-2) \istb  \endist
\istroot(A2)(2-1) \istb \istb  \endist
\cntmistb
\istrootcntm(3)(A2-2) \istb  \endist
\istroot(A3)(3-1) \istb \istb  \endist
\cntmistb
\istrootcntm(4)(A3-2) \istb  \endist
\istroot(A4)(4-1) \istb \istb  \endist
\end{istgame}
\end{lstlisting}

In the example below, \texttt{xtHideEndPoints} and \texttt{xtHidePoints*} cooperatively turn off the effects of \texttt{xtShowEndPoints*} and \texttt{cntmAistb*}.

\begin{lstlisting}[language=istgame]
\begin{istgame}
\cntdistance*{8mm}{16mm}
\xtShowEndPoints*[oval node]
\cntmAistb*
\istrootcntmA(1)  \istbA  \endist
\istroot(A1)(1-1) \istbA \istbA  \endist
\xtShowEndPoints*[box node]
\cntmAistb
\istrootcntmA(2)(A1-2) \istbA  \endist
\istroot(A2)(2-1) \istbA \istbA  \endist
\cntmAistb
\istrootcntmA(3)(A2-2) \istbA  \endist
\istroot(A3)(3-1) \istbA \istbA  \endist
\cntmAistb*
\xtHideEndPoints* \% turns off only cntm endpoints
\xtHideEndPoints
\istrootcntmA(4)(A3-2) \istbA  \endist
\istroot(A4)(4-1) \istbA \istbA  \endist
\end{istgame}
\end{lstlisting}
11.5 Information sets for continua of branches

11.5.1 Information sets for continuum triangles

It is not an issue to draw information sets for continuum triangles. You can just apply \xtInfoset or \xtInfoset0.

\begin{istgame}
\cntmdistance*{8mm}
\cntmpreset{1}[green]
\istrootcntm(1) \istb \endist
\xtInfoset([yshift=-1mm]cntm-1)([yshift=-1mm]cntm-2)
\istroot(A1)(1-1) \istb \istb \endist
\xtInfoset0([fill=red,opacity=.4](cntm-1)(cntm-2)
\istroot(A2)(2-1) \istb \istb \endist
\end{istgame}

11.5.2 \cntmAInfoset for continuum arcs

The macro \cntmAInfoset draws an arc to represent an information set for an arc type continuum of branches, on the background layer. The default position of the arc information set is 1+.1, meaning that .1 (by default) is added to the position (1 by default) of a continuum arc. It must be used right after the corresponding \istrootcntmA whose root coordinate is its mandatory argument. It does not have an option for an information set owner. (If needed, you can use supplementary macros such as \xtInfosetOwner and \xtOwner.)

% syntax: \cntmAInfoset
\cntmAInfoset*[<line opt>]{<cntm root>}{plot factor>{<add-to cntmApos>}
% defaults: (<cntm root>) is mandatory
[-,samples=500,tension=1,infoset style](<m>)<1.333>{.1}

You can change, if necessary, the curvature of an information set arc by using plot factor (<1.333> by default).

% \cntmAInfoset
\begin{istgame}[scale=1.5]
\istrootcntmA(0) \istbA \endist
\xtInfoset(0){1.34>
\xtdistance{10mm}{20mm}
\istroot(1)(0-1) \istb \istb \endist
\end{istgame}

You can also change the position and the line style of an information set arc, as shown in the following example:

% Example: \cntmAInfoset
\begin{istgame}[scale=1.5]
\istrootcntmA(0) \istbA \endist
\cntmAInfoset[red,dashed,thick](0){-.1}
\cntmAInfoset(0)
\cntmAInfoset[blue,dashed,thick](0){.2}
\xtdistance{10mm}{20mm}
\istroot(1)(0-1) \istb \istb \endist
\end{istgame}
11.5.3 \texttt{cmtmAInfosetO}: oval version

The oval version \texttt{cmtmAInfosetO} enables you to draw a bubble type information set for a continuum arc. It must be used right after a simple tree with the root produced by \texttt{istrootcmtmA} and accepts the root of the continuum as the mandatory argument. \texttt{cmtmAInfosetO} can control the style of the information set and height (or thickness), but it does not have an option for the owner of an information set.

\begin{verbatim}
\% syntax: \texttt{cmtmAInfosetO}
\texttt{cmtmAInfosetO}[<line opt>](<cmtm root>)
\texttt{cmtmAInfosetO}[<plot factor>][<add-to cmtm Apos>][<turn X-ang>](<infoset height>)
\% defaults: (<cmtm root>) is mandatory
[-,samples=500,tension=1,infoset style](<m>)<1.333>{.1}[0](1em)
\end{verbatim}

You can change the style of an information set by using usual \textsc{tikz} options. You can also change its position and curvature via \texttt{cmtmA preset}.

\begin{verbatim}
% \texttt{cmtmAInfosetO} and \texttt{cmtmA preset}<plot factor>
\begin{istgame}[scale=1.2]
\cmtmdistance*(10mm){20mm}
\texttt{cmtmAInfosetO}[blue,dashed,thick](0)
\cmtmA preset(.6)
\istrootcmtmA(1)(0-1) \istbA \endist
\texttt{cmtmAInfosetO}[solid,fill=green](1)(.5em)
\cmtmA preset(1.8){.6}
\istrootcmtmA(2)(1-1) \istbA \endist
\texttt{cmtmAInfosetO}[fill=blue!20](2)
\end{istgame}
\end{verbatim}

You do not need to bother if a tree is swapped or rotated. Note also that the height (or thickness) of information set are not affected by \textsc{tikz} scaling.

\begin{verbatim}
% \texttt{cmtmAInfosetO}
\begin{istgame}[scale=.8]
\setistgrowdirection'{east}
\cmtmdistance*(10mm){20mm}
\texttt{cmtmAInfosetO}[blue,dashed,thick](0)
\cmtmA preset(.6)
\istrootcmtmA(1)(0-1) \istbA \endist
\texttt{cmtmAInfosetO}[solid,fill=green](1)(.5em)
\cmtmA preset(1.8){.6}
\istrootcmtmA(2)(1-1) \istbA \endist
\texttt{cmtmAInfosetO}[fill=blue!20](2)
\end{istgame}
\end{verbatim}

If necessary, you can use the \texttt{turn X-angle} option to get a better result (see the following Section 11.5.4 for this issue).
11.5.4 Fine-tuning \cntmAInfosetO: Not for most users

turing X circles

In the case of a severely asymmetrical \texttt{TikZ} scale (using \texttt{xscale} and \texttt{yscale} asymmetrically), an information set drawn by \texttt{\cntmAInfosetO} will probably be distorted, as shown in the upper part of the example below. The bracket option to be used after the mandatory argument of \texttt{\cntmAInfosetO} can solve this problem by turning the blue circle (called, an X circle) drawn at the beginning point of the information set. To do that, specify an appropriate degree of rotation in brackets, like \texttt{\cntmAInfosetO(2)[60]}, as shown below. (See Section 10.5.1 for a related topic.)

\begin{lstlisting}
\begin{istgame}[xscale=4]
\cntmdistance*{15mm}{10mm}
\cntmApreset<1.8>
\istrootcntmA(1) \istbA \endist
\node at (cntm-1)
  [draw,circle,blue,opacity=.4,minimum size=1em]{};
\cntmAInfosetO(1)
\istrootcntmA(2)(1-1) \istbA \endist
\cntmAInfosetO(2)[60]
\end{istgame}
\end{lstlisting}

scaling \texttt{\cntmAInfosetO} according to \texttt{TikZ} scales

The height (or thickness) of an information set drawn by \texttt{\cntmAInfosetO} is not affected by \texttt{TikZ} scales. In some cases, you may want to change the height according to the values of \texttt{TikZ} scales.

Remark:

- As discussed in Section 10.5.2, the \texttt{istgame} environment treats, for example, \texttt{[scale=.7]} and \texttt{[scale=.7,xscale=1]} as different, while the \texttt{tikzpicture} environment treats them as equal.

- With the package \texttt{istgame}, if you use \texttt{[scale=.7,xscale=1]} then the value of \texttt{scale} is not used to keep the height of an oval type information set unchanged, so that you can scale the information set accordingly.

\begin{lstlisting}
\begin{istgame}[scale=.7,xscale=1]
\istrootcntmA(0)
\istbA
\endist
\cntmAInfosetO[fill=blue!20](0)
\xtdistance{10mm}{20mm}
\istroot(1)(0-1)
\istb
\istb
\endist
\end{istgame}
\end{lstlisting}

11.6 Transition from obsolete macros \texttt{\lstcntm} and \texttt{\lstcntmarc}

It turned out to be inconvenient to use the two macros \texttt{\lstcntm} and \texttt{\lstcntmarc} to express a continuum of branches because it requires two steps. Use, instead, a one step method (if you use only default settings) that combines the two steps with \texttt{\istrootcntm} and \texttt{\istrootcntmA}. Anyway, the two obsolete macros are provided only for backward compatibility with no maintenance and possible to be removed later.
The macro \istcntm draws only a background triangle for a continuum of branches. And then you can draw an action taken using \istroot.

\begin{verbatim}
\istcntm
\[\text{syntax:}\]
[\text{grow keyval}](<coor1>)(<coor2>)[<fill color>]+<levdist>..<sibdist>+
\[\text{defaults:}\]
[south](){(0,0)[black!25]+8mm..24mm+}
\end{verbatim}

The above old way can be switched to the following new way:

\begin{verbatim}
\istrootcntm
[\text{syntax:}\]
[\text{grow keyval}](<coor1>)(<coor2>)\{<num>\}+levd..sibd+
\[\text{defaults:}\]
[south](){(0,0)[bend right]{.5}+8mm..24mm+}
\end{verbatim}

The above old way can be switched to the following new way. One difference is that the position of a continuum arc is now 1 by default, while it was .5.
12 Arrows

12.1 Using Ti\textsf{K}Z arrow options with \texttt{istgame}

12.1.1 Ti\textsf{K}Z arrow option $\rightarrow$

The simplest way of printing arrows in \textit{a whole game tree} is to use the Ti\textsf{K}Z option $\rightarrow$ as an option for the \texttt{istgame} environment. To get the best result, you may want to use \texttt{\xtShowEndPoints} together with the option $\rightarrow$.

For an arrow on an individual branch, you can use arrow options with each \texttt{\istb}.

12.1.2 Fine-tuning with \texttt{\setistgameshorten}: Not for most users

Assuming that you do not want the terminal points to be shown, in order to get the best result of arrows you can use the macro \texttt{\setistgameshorten}, like \texttt{\setistgameshorten{1.3pt}}, right \textit{before} the \texttt{istgame} environment. Note that \texttt{\setistgameshorten} has a global effect. So, when necessary, you should use the macro within a Te\textsf{X} group.

\textbf{Remark:} (not for most users)

- Internally, the \texttt{istgame} environment checks the existence of the option $\rightarrow$ and automatically adds \texttt{shorten >=<keyval>} (by default, 0pt) to the option list as the first entry together with $\rightarrow$.
- You can do the same thing, without using this macro, by manually adding, for example, $\rightarrow$, \texttt{shorten >=1.3pt} to the list of options.
You should also be aware that, in TikZ, the options `shorten` and `->` affect any lines or curves in the current environment.

12.2 Middle arrows for a branch

With `\istb`, you can print an arrow tip in the middle of a branch by using a new `controllable` arrow tip style `->-`, like `\istb[->-].`

12.2.1 A middle arrow for each `\istb`

To print an arrow tip in the middle of a branch, you can use `->-` as an option for `\istb`. The `controllable` middle arrow style `->-` takes one optional value, which is a number between 0 and 1 (by default `->-=.55`) determining a relative position from a parent node to a child node, where the end of an arrow tip is placed. For example, `\istb[->-=.1]` prints an arrow tip near the parent node, and `\istb[->-=.9]` near the child node.

### Remark: issues on middle arrows with `\istB`:

- Middle arrows work well with `\istb`, but the exact position of a middle arrow tip does NOT work very well with `\istB`.
- So, if you want to print dual action labels for branches with middle arrows, it is recommended to use `\istb` together with `\xtActionLabel`, instead of `\istB` (see also page 28).

Except for `\istB`, all the other variants of `\istb` including `\istbA`, `\cntmistb`, and `\cntmAistb` work well with the middle arrow option `->-`.

12.2.2 Middle arrow tip styles

The package also provides additional styles for middle arrow tips, which have a `fixed` position and shape. (Note that the single arrow tip `->--` is `controllable`.)

Additional middle arrow tip styles:

- `->>>-` : double arrow tip in the middle of a branch at the position of .6
- `-------------` : triple arrow tip in the middle of a branch at the position of .65
- `--o--` : circle arrow tip in the middle of a branch at the position of .55
- `--x--` : cross arrow tip in the middle of a branch at the position of .5
Remark: All the middle arrow tips are printed in black, by default. Their color can be changed by \texttt{\textbackslash setxtarrowtips} (see the following Section 12.2.3).

\begin{verbatim}
% Example: middle arrow tips
\begin{istgame}[xscale=.8]
\istroot(0)+15mm..30mm+
\istb[->-,blue]
\istroot(1)(0-1)
\istb[-o-,red]
\endist
\setxtarrowtips[blue,thick]
\istroot(2)(0-2)
\istb[-o-]
\endist
\istroot(3)(1-2)
\istb[->->-,red]
\istroot(4)(2-2)
\istb[-o-]
\endist
\end{istgame}
\end{verbatim}

12.2.3 \texttt{\textbackslash setxtarrowtips}: controlling middle arrow tips via \texttt{\textbackslash ->-}

The macro \texttt{\textbackslash setxtarrowtips} controls the position and shape of a middle arrow tip, but only works through the \textit{controllable} middle arrow style \texttt{\textbackslash ->-}, not the other \textit{fixed} arrow styles. However, it controls the color of all the middle arrow tips. The effect of change by this macro continues until the end of the current environment unless it is changed again. To restore the default values, just declare \texttt{\textbackslash setxtarrowtips} followed by nothing.

\begin{verbatim}
% \setxtarrowtips
% syntax: all arguments are optional
% \setxtarrowtips<midarrow tip pos>{<arrow tip shape>}[<arrow opt>]
% \setxtarrowtips
\begin{verbatim}
% \setxtarrowtips
% \begin{istgame}[scale=1]
% \setxtarrowtips<.7>{Bar}[blue,thick]
\istroot(0)+15mm..25mm+
\istb[->-]
\istb[->-]
\endist
\setxtarrowtips<.8>{Rays}[red,thick]
\istb[->-,green]
\istb[->-,red]
\endist
\setxtarrowtips<.7>
\istroot(3)(1-2)
\istb[->->-]
\istroot(4)(2-2)
\istb[-o-]
\endist
\end{istgame}
\end{verbatim}
\end{verbatim}

Note that, as seen in the example above, the position set individually by an option value of \texttt{\textbackslash istb}, like \texttt{\istb[->-=.3]}, wins the position set by \texttt{\setxtarrowtips<.9>}. 

12.3 Middle arrows for simple trees

12.3.1 \texttt{\textbackslash xtShowMidArrows} and \texttt{\textbackslash xtHideMidArrows}

The macro \texttt{\textbackslash xtShowMidArrows} prints \textit{middle arrows} (a single arrow tip, by default) for \textit{all the branches} of a \textit{simple tree} (but not for a separate branch). The effect of \texttt{\textbackslash xtShowMidArrows} continues until it is changed by another \texttt{\textbackslash xtShowMidArrows} or blocked by \texttt{\textbackslash xtHideMidArrows}.
Remark:

- \(\texttt{\xtShowMidArrows}\) works for all branches of simple trees.
- The middle arrow option styles work separately only for \(\texttt{\istb}'\)s having them and \(\texttt{\setxtarrowtips}\) works only for \(\texttt{\istb}'\)s having the option \(-\rightarrow\).  
- Since the middle arrow tip styles and \(\texttt{\xtShowMidArrows}\) work completely independently, if you use both of them, they are all printed.
- Note also that \(\texttt{\xtHideMidArrows}\) removes only the effect of \(\texttt{\xtShowMidArrows}\).

12.3.2 \(\texttt{\setxtshowmidarrows}\): controlling \(\texttt{\xtShowMidArrows}\)

The macro \(\texttt{\setxtshowmidarrows}\), you can control the position, color, style, and shape of middle arrow tips, to be drawn by \(\texttt{\xtShowMidArrows}\).
Remark: issues in adding other graphic objects (not for most users)

- As you can see in the above example, when `\xtShowMidArrows` is used, you should redefine coordinates for endpoints before you use them for other graphic objects.

- This is because, in Ti\kZ, `shift` or `scope` does not work in this case. So, alternatively, you can use `\draw [transform canvas={yshift=-5pt}] (2-1) -- (2-2);` (See Ti\kZ manual about `transform canvas`).

- `\xtShowArrows` does not affect the Ti\kZ absolute coordinates.

12.4 Arrows for simple trees

12.4.1 `\xtShowArrows` and `\xtHideArrows`

In many cases you might not need to care about the position of arrow tips, just wanting to put them at the ends of branches for simple trees. For this purpose, the macro `\xtShowArrows` (accepting one bracket option for a node style) is provided.

`\xtShowArrows`, internally, calls `\xtShowEndPoints` (with `solid node`, by default) first and then prints arrows in order to produce the best result of showing arrows. Therefore, for example, `\xtShowArrows[oval node]` is equivalent to `\xtShowEndPoints[oval node]` followed by printing arrows (in black by default).

The macro `\xtHideArrows` turns off all the effect of `\xtShowArrows`, while `\xtHideArrows*` has the endpoints remained. (`\xtShowArrows` does not have its starred version.)
Remark: issues in adding other graphic objects (not for most users)

- Like \texttt{\textbackslash xtShowMidArrows}, when \texttt{\textbackslash xtShowArrows} is used, you should redefine coordinates for endpoints before you use them for other graphic objects, or use \texttt{transform canvas}.

12.4.2 \texttt{\setxtshowarrows}: controlling \texttt{\xtShowArrows}

The macro \texttt{\setxtshowarrows} controls the color and shape of arrow tips. You can even change the position of arrow tips by specifying the distance (1.2pt by default) from the endpoint as the last parenthesis optional argument, like (5pt).

% \setxtshowarrows
% syntax:
% \setxtshowarrows{<arrow shape>}[<arrow opt>](<arrow end shorten dim>)
% defaults: {>}[](1.2pt)

\begin{istgame}
\setxtshowarrows{Stealth[open]}[blue](5pt)
\xtShowArrows
\istroot(0)(0,0){1}
\istb{x}[r] \istbm \endist
\xtHideArrows
\istroot(1)(0-1)
\istb{a} \istb \endist
\xtHideArrows* % endpoints remain
\istroot(2)(0-2)
\istb{b} \istb \endist
\xtShowArrows[oval node]
\istroot(3)(1-2)
\istb{c} \istb \endist
\end{istgame}

12.4.3 Arrows and continua of branches: examples (Not for most users)

There is nothing special in using arrow tips in game trees having continua of branches.

% Example: cntm and arrows
\begin{istgame}[font=\scriptsize]
\xtdistance{10mm}{20mm}
\xtShowArrows
\istrootcntm(0,0){1}
\istb{x}[r] \istbm \endist
\istroot(1)(0-1)<[label distance=-3pt]120>{2}
\istb{Y}[al]{x,1-x} \istb{N}[ar] \endist
\istrootcntm(2)(1-2){2}
\istb{y}[r] \istbm \endist
\istroot(3)(2-1)<[label distance=-3pt]120>{1}
\istb{Y}[al]{1-y,y} \istb{N}[ar] \endist
\end{istgame}

It is not impossible to put arrow tips on the outermost branches of a continuum. To do that, you can use \texttt{\textbackslash cntm\textbackslash istb(*)} and \texttt{\textbackslash cntmA\textbackslash istb(*)} together with the middle arrow tip styles and \texttt{\setxtarrowtips}. 

82
Here is one more example.

\begin{istgame}
\setistarrowtips
\xtowner<.93>{>>}[red,thick]
\cntmAistb[->-]
\istrootcntmA(1)
\istbA[-x-] \endist
\istrootcntmA(2)(A1-2) \istb \istb \endist
\istrootcntmA(3)(A2-2) \istbA[-x-] \endist
\istrootcntmA(4)(A3-2) \istbA[-x-] \endist
\istrootcntmA(5)(A4-4) \istb \istb \endist
\end{istgame}

13 Supplementary macros

13.1 Supplementary macros to important labels

The \texttt{istgame} package also provides some supplementary macros to important labels: players, action labels, and payoffs. The supplementary macros work outside of simple trees.

\texttt{\xtowner} puts the owner of a node (or a player), in text mode
\texttt{\xtowner*} puts an owner in the input mode as set by \texttt{\setistmathTF(*)}

\texttt{\xtactionlabel} puts an action label to a branch, in math mode
\texttt{\xtactionlabel*} puts an owner in the input mode as set by \texttt{\setistmathTF(*)}
\texttt{\xtPayoff} puts payoffs, in math mode
\texttt{\xtPayoff*} puts payoffs in the input mode as set by \texttt{\isetmathTF(*)}

\begin{verbatim}
\syntax: \xtPayoff\{<coor>\}{<payoff>}[<pos>,<node opt>]
\defaults: \{}\{}[\text below,\text inner sep=0pt,\text outer sep=4pt,\text text depth=.25ex]
\end{verbatim}

(abbreviations for \texttt{pos} available)

\texttt{\xtInfosetOwner} puts the owner of an information set, in text mode
\texttt{\xtInfosetOwner*} puts an information set owner in the input mode as set by \texttt{\isetmathTF(*)}

\begin{verbatim}
\syntax: \xtInfosetOwner\{<from>\}\{<to>\}\{<owner>\}[<pos>,<node opt>]
\defaults: \{\}\{}\{}
\end{verbatim}

(abbreviations for \texttt{pos} available)

\texttt{\xtNode} puts ‘something,’ say, an owner into a node
\texttt{\xtNode*} puts ‘something’ into a plain node

\begin{verbatim}
\syntax: \xtNode\{<coor>\}\{<node style>,<node opt>\}\{<node text>\}
\syntax: \xtNode*\{<coor>\}\{<node style>,<node opt>\}\{<node text>\}
\defaults: \{\}\{\}\{}
\end{verbatim}

The supplementary macros depend on the coordinates defined in the sequence of \texttt{\istroot–\istb–\endist} and print their corresponding objects on or around the specified coordinates. So, in the above supplementary macros, all the coordinates such as \{\texttt{from}\}, \{\texttt{to}\}, and \{\texttt{coor}\} are mandatory arguments, denoted by \texttt{m}.

% Example: various supplement macros
\begin{istgame}[scale=3]
\istroot\{0\}\{chance node\}
\istb\{dashed\}\{A\}\ \istb\{blue,very thick\}\{B\}\{right,xshift=2pt,yshift=2pt,red\}
\endist
\xtInfoset\{0\}\{0\}
\xtInfoset\{thick\}\{0-1\}\{0-2\}\{1\}
\xtInfoset\{thick,red,rounded corners=1em\}\{0-2\}\{0-3\}\{2\}\{2em\}
\xtInfoset\{dashed,bend right=25\}\{0-1\}\{0-3\}
\xtActionLabel\{0\}\{0-3\}\{1-\epsilon\}\{below,sloped\} % action label in math mode
\xtInfosetOwner\{0\}\{0-3\}\{3\}\{xshift=-25pt,yshift=-25pt\}
\xtOwner\{0\}\{Nash\}\{xshift=-5pt,left\}
\xtPayoff\{0-3\}\{(u_1,u_2)\}\{right,xshift=5pt\} % payoffs in math mode
\xtNode\{dotted\}\{0-1\}\{box node,fill=blue!20\}\{Smith\}
\xtNode*\{a-1\}\{text\}
\end{istgame}
To specify the [<pos>] option for the above supplementary macros (other than \xtNode), you can use the abbreviations [1], [x], [a], [b], [al], [ar], [bl], and [br]. Each of these abbreviations works only when used alone without any other option keys in the option list. For example, the option [b] or [below] or [below,xshift=5pt] works, but [b,xshift=5pt] does not, producing a compile error.

13.2 More supplementary macros

13.2.1 \xtTimeLineH and \xtTimeLineV

With the macro \xtTimeLineH, you can draw a horizontal time-line.

\begin{istgame}
\setistgrowdirection{south} % default
\xtdistance{10mm}{20mm}
\istrootcntm(1)
\istroot(A1)(1-1)
\istrootcntm(2)(A1-2)
\istroot(A2)(2-1)
\istrootcntm(3)(A2-2)
\istroot(A3)(3-1)
\xtTimeLineH[->](1){-1.5}{3}{period 1}
\xtTimeLineH(2){-1.5}{3}{period 2}
\xtTimeLineH'(3){-1.5}{3}{period 3}
\end{istgame}

There are also \xtTimeLineV for vertical time-lines.

Remark: The direction to which a time-line goes depends on whether a tree is swapped or not.

- If branches are arranged counterclockwise (by default), a time-line goes from left to right or from below to above. As to \xtTimeLineH, we take this as a standard case: left to right.
- If a tree is swapped, so the branches are arranged clockwise, \xtTimeLineH draws a line going to the opposite direction: from right to left (for trees growing southwards). So, if a tree is swapped by \setistgrowdirection, for example, \xtTimeLineH(coor){-1.5}{3} draws a time-line going to the opposite direction: from 1.5cm to -3cm.

The following codes are the same as the above, except that the tree is swapped.
The following codes are the same as the above, except that the tree is not swapped.

In the example above, since the tree is not swapped, \texttt{\textbackslash xtTimeLineV(coor)\{1.5\}\{-3\}} draws a vertical time-line going to the opposite direction: from -1.5cm to 3cm.
13.2.2 \texttt{\textbackslash CommentTo} and \texttt{\textbackslash CommentFrom}

With the macro \texttt{\textbackslash CommentTo}, you can attach a comment to a node from a coordinate relative to the node with a densely dotted arrow line (by default). Note that with \texttt{\textbackslash CommentTo} you need to specify a target absolute coordinate followed by a start relative coordinate.

\begin{verbatim}
% \texttt{\textbackslash CommentTo}
% \textit{syntax:}
% \texttt{\textbackslash CommentTo}[\langle line opt\rangle]\langle absolute to-coor\rangle\langle relative from-coor\rangle\{\langle comment\rangle\}
% [\langle pos\rangle,\langle node opt\rangle]\langle end-shorten dim\rangle
% \textit{defaults:}
% [densely dotted,->,>=stealth,shorten >=1pt]\langle m\rangle\langle m\rangle\{\}\{above\}(1pt)
\end{verbatim}

The last parenthesis option if for shortening arrow end. You can use the abbreviations of the directional words, but not with other options in the option list.

\begin{verbatim}
% \texttt{\textbackslash CommentTo}
\begin{istgame}[->,scale=.7]
\xtShowEndPoints
\istroot(0)+15mm..30mm+ \istb \istb \endoist
\istroot(1)(0-1) \istb \istb \endoist
\istroot(2)(0-2) [chance node] \istb \istb \endoist
\xtCommentTo[bend right](0)(150:2){Alice moves}[l]
\xtCommentTo[bend right](1)(150:2){Ben moves}
\xtCommentTo[bend left](2)(0,2){Nature moves}[a](5pt)
\end{istgame}
\end{verbatim}

The macro \texttt{\textbackslash CommentFrom} enables you to attach a comment to one or more nodes from an absolute coordinate with a densely dotted arrow line. Note that with \texttt{\textbackslash CommentFrom} you need to specify a start absolute coordinate followed by a target absolute coordinate. The last parenthesis option if for shortening arrow end. You can use the abbreviations of the directional words.

\begin{verbatim}
% \texttt{\textbackslash CommentFrom}
% \textit{syntax:}
% \texttt{\textbackslash CommentFrom}[\langle line opt\rangle]\langle from\rangle\langle to\rangle\{\langle comment\rangle\}[\langle pos\rangle,\langle node opt\rangle]\langle arrow end shorten\rangle
% \textit{defaults:}
% [densely dotted,->,>=stealth,shorten >=1pt]\langle m\rangle\langle m\rangle\{\}\{above\}(1pt)
\end{verbatim}

\begin{verbatim}
% \texttt{\textbackslash CommentFrom}
\begin{istgame}[scale=.6]
\xtShowEndPoints
\istroot(0)+15mm..30mm+ \istb \istb \endoist
\istroot(1)(0-1) \istb \istb \endoist
\istroot(2)(0-2) [chance node] \istb \istb \endoist
\xtCommentFrom[bend right](-3,2)(0){decision nodes}
\xtCommentFrom[bend right](-3,2)(1)
\xtCommentFrom[bend right,solid,blue](-3,2)(2)
\xtCommentFrom[bend right](-3,-5)(1-1){terminal nodes}[b]
\xtCommentFrom[bend right](-3,-5)(1-2)
\xtCommentFrom[bend right](-3,-5)(2-1)
\xtCommentFrom[bend right](-3,-5)(2-2)(5pt)
\end{istgame}
\end{verbatim}
14 Representing subgames

Here is an example of a whole game tree.

\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<O>(1)
\istb(A)[a]{{(2,2)}}{1} \istb(D)[r] \endist
\istroot(1)(0-2)<135>{1}
\istb(L)[al] \istb(R)[ar] \endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
\istb{\ell}[al]{{(4,2)}} \istb{r}[ar]{{(1,1)}} \endist
\istroot(3)(1-2)<45>{2}
\istb{\ell}[al]{{(3,2)}} \istb{r}[ar]{{(0,3)}} \endist
\xtSubgameBox(1){(2-1)(3-2)}[solid,blue]
\xtSubgameBox(2){(2-1)(2-2)}[solid,blue]
\end{istgame}

14.1 \xtSubgameBox (experimental)

\xtSubgameBox is for indicating a subgame by a box (or a rectangle) with rounded corners.

The subgame box embraces the subroot indicated in parentheses and terminal nodes specified within curly braces. The coordinates of terminal nodes are listed in curly braces without delimiters (spaces are allowed). The default options for the subgame box are \textit{dashed}, \textit{inner sep=20pt}, \textit{rounded corners=15pt}, and \textit{black!50}. 

\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<O>(1)
\istb(A)[a]{{(2,2)}}{1} \istb(D)[r] \endist
\istroot(1)(0-2)<135>{1}
\istb(L)[al] \istb(R)[ar] \endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
\istb{\ell}[al]{{(4,2)}} \istb{r}[ar]{{(1,1)}} \endist
\istroot(3)(1-2)<45>{2}
\istb{\ell}[al]{{(3,2)}} \istb{r}[ar]{{(0,3)}} \endist
\xtSubgameBox(1){(2-1)(3-2)}[solid,blue]
\xtSubgameBox(2){(2-1)(2-2)}[solid,blue]
\end{istgame}
You can change the size of the box in two ways. First, you can change it by using `inner sep`, `inner xsep`, or `inner ysep`. Secondly, you can shift the subroot, like \([xshift=5pt]1\) when the subroot is \((1)\). Of course, you can do the both at the same time.

\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot{-135}[0]{initial node}<0>{1}
\istb{A}[a]{(2,2)}[l] \istb{D}[r]\endist
\istroot{1}[0-2]<135>{1}\istb{L}[a]\istb{R}[a]\endist
\xtdistance{10mm}{20mm}
\istroot{2}[1-2]<135>{1}
\istb{L}[al]
\istb{R}[ar]\endist
\xtdistance{10mm}{20mm}
\istroot{2}[1-2]<135>{2}
\istb{L}[al]
\istb{R}[bl]\endist
\xtdistance{10mm}{20mm}
\istroot{3}[1-2]<-135>{2}
\istb{L}[al]
\istb{R}[bl]\endist
\xtSubgameBox*[\[xshift=20pt\]1]{(2-1)(3-2)}% [inner xsep=30pt]
\end{istgame}

14.1.2 \xtSubgameBox*

The starred version `\xtSubgameBox*` produces a subgame box with background color (by default red!20). In the following example, where the tree grows east, the size of the filled subgame box is adjusted by manipulating `inner sep` and `xshift`.

\begin{istgame}
\setistgrowdirection'\{east\}
\xtdistance{15mm}{30mm}
\istroot{45}[0]{initial node}<180>{1}
\istb{A}[a]{(2,2)}[l] \istb{D}[b]\endist
\istroot{1}[0-2]<135>{1}\istb{L}[a]\istb{R}[b]\endist
\xtdistance{10mm}{20mm}
\istroot{2}[1-2]<135>{2}
\istb{L}[al]
\istb{R}[bl]\endist
\istroot{3}[1-2]<-135>{2}
\istb{L}[al]
\istb{R}[bl]\endist
\xtSubgameBox*[\[xshift=20pt\]1]{(2-1)(3-2)}% [inner xsep=30pt]
\end{istgame}

14.2 `\xtSubgameOval and `\xtSubgameOval*` (experimental)

`\xtSubgameOval` and `\xtSubgameOval*` are also provided.

% `\xtSubgameOval(*)`
% syntax:
% `\xtSubgameOval{(*)\langle\subroot\ coor\rangle\{\langle\coor1\rangle\ \langle\coor2\rangle\ ...\ \}\langle\opt\rangle}`
% options:
% *: filled box
% \langle\subroot\ coor\rangle: the subroot of a subgame (mandatory)
% \{\langle\coor1\rangle\ \langle\coor2\rangle\ ...\ \}: coordinates of terminal nodes (mandatory)
% [\langle\opt\rangle]: color, line style
% defaults:
% \{\langle\m\rangle\}\{ellipse,dashed,inner sep=20pt,rounded corners=15pt,black!50\}
% *\{\langle\m\rangle\}\{ellipse,inner sep=20pt,rounded corners=15pt,red!20\}
\texttt{\textbackslash xtSubgameOval(*)} works just like \texttt{\textbackslash xtSubgameBox(*)}, except for one thing: an oval instead of a rectangle. \texttt{\textbackslash xtSubgameOval} draws an oval (or an ellipse) to represent a subgame, and \texttt{\textbackslash xtSubgameOval*} draws a filled oval with red!20 by default.

The way of changing the size of the subgame oval is the same as with \texttt{\textbackslash xtSubgameBox}: first, changing inner sep, and secondly, shifting the subroot. In the example below, the options \texttt{inner xsep=25pt} and \texttt{[xshift=30pt]1} are used to change the size of the subgame oval. Its color is changed by \texttt{blue!20}.

\begin{Verbatim}
\% \texttt{\textbackslash xtSubgameOval}
\begin{istgame}
\xdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<0>{1}
  \istb(A)[a]{(2,2)}[1]
  \istb(D)[r]
\endist
\istroot(1)(0-2)<135>{1}
  \istb(L)[al]
  \istb(R)[ar]
\endist
\xdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
  \istb(\ell)[al]{(4,2)}
  \istb(r)[ar]{(1,1)}
\endist
\istroot(3)(1-2)<45>{2}
  \istb(\ell)[al]{(3,2)}
  \istb(r)[ar]{(0,3)}
\endist
\xtSubgameOval(1){(2-1)(3-2)}
\xtSubgameOval(2){(2-1)(2-2)}
\end{istgame}
\end{Verbatim}

\begin{Verbatim}
\% \texttt{\textbackslash xtSubgameOval*}
\begin{istgame}
\setistgrowdirection'{east}
\xdistance{15mm}{30mm}
\istroot[45](0)[initial node]<180>{1}
  \istb(A)[a]{(2,2)}[a]
  \istb(D)[b]
\endist
\istroot(1)(0-2)<135>{1}
  \istb(L)[al]
  \istb(R)[ar]
\endist
\xdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
  \istb(\ell)[al]{(4,2)}
  \istb(r)[ar]{(1,1)}
\endist
\istroot(3)(1-2)<45>{2}
  \istb(\ell)[al]{(3,2)}
  \istb(r)[ar]{(0,3)}
\endist
\xtSubgameOval*[(xshift=30pt)]1{(2-1)(3-2)}\%[inner xsep=25pt,blue!20]  
\end{istgame}
\end{Verbatim}
14.3 \setxtsubgamelayer

You can change the shape of a subgame representation to other shapes such as \texttt{semicircle} and \texttt{circle} or even \texttt{star}. You will need to change the size of the shape by using \texttt{inner sep} and by shifting the subroot to get the result you want.

% subgame: half circle
\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<>{1}
\istb{A}[a]{{(2,2)}}{1}
\istb{D}[r] \endist
\istroot(1){(0-2)<135>{1}}
\istb{L}[al]
\istb{R}[ar] \endist
\xtdistance{10mm}{20mm}
\istroot(2){(1-1)<135>{2}}
\istb{ell}[al]{{(4,2)}}
\istb{r}[ar]{{(1,1)}}
\endist
\xtSubgameBox((yshift=-2cm)1){(2-1)(3-2)}%[semicircle,inner sep=15pt]
\end{istgame}

Here are more examples showing how to draw a rectangle and circles to represent subgames.

% subgame: circle
\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<>{1}
\istb{A}[a]{{(2,2)}}{1}
\istb{D}[r] \endist
\istroot(1){(0-2)<135>{1}}
\istb{L}[al]
\istb{R}[ar] \endist
\xtdistance{10mm}{20mm}
\istroot(2){(1-1)<135>{2}}
\istb{ell}[al]{{(4,2)}}
\istb{r}[ar]{{(1,1)}}
\endist
\xtSubgameBox*(1){(2-1)(3-2)}\xtSubgameOval*{2}{(2-1)(2-2)}%[circle,blue!20,inner xsep=9pt]
\xtSubgameOval{3}{(3-1)(3-2)}%[circle,inner xsep=9pt]
\end{istgame}

You can use \texttt{\setxtsubgamelayer} to change the layer of a subgame from \texttt{background} to \texttt{behind}, \texttt{main}, \texttt{above}, or \texttt{foreground}, in that order. In order to go back to the default layer, you can just declare \texttt{\setxtsubgamelayer} or do \texttt{\setxtsubgamelayer{}}.
15 Miscellany

15.1 Various branch types, directions, and lengths

Here is an example of how to deal with various line types, directions, and lengths of branches. You can do this using \istbA (see Section 5.5, for more details).
15.2 Code reuse

15.2.1 Drawing subgames

Suppose that you have the following tree as a whole game.

It is easy to draw a subgame if you use the existing codes for a whole game. To get a subgame, just comment out the codes for the rest of the subgame.

Remark: If you copy and paste the previous codes alone and try to compile to get a subgame, you will get an error message:

! Package pgf Error: No shape named 0-2 is known.

This is because the first active \texttt{\textbackslash istroot} refers to the undefined coordinate (0-2). In this case you need to change the line as:
\texttt{\textbackslash istroot(1)(0,0)<left>{1}}
However, if you have the codes for a whole game before the codes with some lines commented out for a subgame, you will not get the error message when you compile the file (because the coordinate (0–2) is already defined in the whole game). Therefore, most of the time you will not get the error message by commenting out some lines to get a subgame.

A smaller subgame is also easy to get by commenting out the rest of the subgame you want.

\begin{istgame}
% Example: a smaller subgame
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<0>{1}
% \istb{A}[a]{{(2,2)}}[1]
% \istb{D}[r]
%\endist
\istroot(1)(0-2)<left>{1}
% \istb{L}[al]
% \istb{R}[ar]
%\endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
\istb{\ell}[al]{{(4,2)}}
\istb{r}[ar]{{(1,1)}}
\endist
\xtdistance{15mm}{30mm}
\istroot{3}(1-2)<45>{2}
\istb{\ell}[al]{{(3,2)}}
\istb{r}[ar]{{(0,3)}}
%\endist
\end{istgame}

It is not possible to make a subgame stay put where it appears in a whole game. (Replace \texttt{black!5} by \texttt{white} to make the rest of the game disappear in the example below.)

\begin{istgame}
% Example: a subgame stays put
\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<0>{1}
\istb{black!5}{A}[a]{{(2,2)}}[1]
\istb{black!5}{D}[r]
%\endist
\istroot(1)(0-2)<left>{1}
\istb{black!5}{L}[al]
\istb{black!5}{R}[ar]
%\endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
\istb{\ell}[al]{{(4,2)}}
\istb{r}[ar]{{(1,1)}}
%\endist
\setistDecisionNodeStyle[black!5]
\istroot{3}(1-2)<45>{2}
\istb{\ell}[al]{{(3,2)}}
\istb{r}[ar]{{(0,3)}}
\endist
\setistDecisionNodeStyle[black!5]
\end{istgame}
15.2.2 Backward induction

It is also easy to illustrate the procedure of backward induction, when you analyze an extensive game.

By using \texttt{\textbackslash istb} with options for the middle arrow tips, you can effectively express the backward induction procedure. The command \texttt{\textbackslash setxtarrowtips} controls the style of arrow tips and just declaring \texttt{\textbackslash setxtarrowtips} followed by nothing restores defaults for the middle arrow tips styles. See Section 12, for more details. change the color and the style of a line representing a branch and an arrow tip to it.

% Example: backward induction
\begin{istgame}[-155][initial node]<0>{1}
  \istb[dashed]{A}[a]{(2,2)}[l]
  \istb[very thick]{D}[r]
\endist
\istroot(0-2)<left>{1}
  \istb[very thick]{L}[al]
  \istb[dashed]{R}[ar]
\endist
\istroot{2}(1-1)<135>{2}
  \istb[very thick]{\ell}[al]{(4,2)}
\istb[thick,dotted]{r}[ar]{(1,1)}
\endist
\istroot(3)(1-2)<45>{2}
  \istb[thick,dotted]{\ell}[al]{(3,2)}
\istb[very thick,blue]{r}[ar]{(0,3)}
\endist
\end{istgame}
15.2.3 Code reusability

Chen (2013)’s work is good to understand how drawing a game tree with TikZ works.

\begin{tikzpicture}[scale=1,font=\footnotesize]
\tikzstyle{solid node}=[circle,draw,inner sep=1.5,fill=black]
\tikzstyle{hollow node}=[circle,draw,inner sep=1.5]
\tikzstyle{level 1}=[level distance=15mm,sibling distance=3.5cm]
\tikzstyle{level 2}=[level distance=15mm,sibling distance=1.5cm]
\tikzstyle{level 3}=[level distance=15mm,sibling distance=1cm]
\node (0)[solid node,label=above:{$P1$}]{}
child{node[solid node,label=above left:{$P2$}]{}
child{node[hollow node,label=below:{$(1,2)$}]{ edge from parent node[left]{$C$}}}
child{node[hollow node,label=below:{$(1,-1)$}]{ edge from parent node[left]{$D$}}}
child{node[hollow node,label=below:{$(0,2)$}]{ edge from parent node[right]{$E$}} edge from parent node[left,xshift=-5]{$A$}}}
child{node[solid node,label=above right:{$P2$}]{}
child{node[hollow node,label=below:{$(2,2)$}]{ edge from parent node[left]{$F$}}}
child{node[hollow node,label=below:{$(1,3)$}]{ edge from parent node[right]{$G$}} edge from parent node[right,xshift=5]{$B$}}};
\end{tikzpicture}

\begin{istgame}[scale=1,font=\footnotesize]
\setistDecisionNodeStyle{4pt}
\xtdistance{15mm}{3.5cm}
\istroot{0}{$P1$}
\istb{A}[al]
\istb{B}[ar]
\xtShowEndPoints[oval node]
\xtdistance{15mm}{1.5cm}
\istroot{1}{0-1}<135>[$P2$]
\istb{C}[1]{(1,2)}
\istb{D}[1]{(1,-1)}
\istb{E}[r]{(0,2)}
\istroot{2}{0-2}<45>[$P2$]
\istb{F}[1]{(2,2)}
\istb{G}[r]{(1,3)}
\end{istgame}

You can get a quite similar (actually the same) result by using the istgame package.

The istgame package enhances simplicity and readability, and hence it is easy to reuse codes.
You can easily read and modify game tree codes even if you go over them after a while.
16 Game tree examples

This section provides some examples of extensive games. Before we start, let us change the default font size of the istgame environment by stating `\setistgamefontsize{\footnotesize}` outside of the environment. (Right here!)

16.1 Simple examples

```latex
\begin{istgame}[->,shorten >=1.3pt]
\setistmathTF*
\xtdistance{15mm}{30mm}
\istroot(0)[initial node]{Child}
\istb{Good}[above left]{(0,2)}
\istb{Bad}[above right]
\endist
\istroot(1)(0-2)<30>{Parent}
\istb{Forgive}[above left]{(1,1)}
\istb{Punish}[above right]{(-1,-1)}
\endist
\end{istgame}
```

% dual sloped action labels (\istB)

```latex
\begin{istgame}
\setistmathTF* % action labels in italics
\xtShowArrows
\xtdistance{15mm}{30mm}
\istroot(0)[initial node]{Child}
\istB{Good}[above,sloped]{$p$}[below, sloped]{(0,2)}
\istB{Bad}[above,sloped]{$1-p$}[below, sloped]
\endist
\istroot(1)(0-2)<30>{Parent}
\istB{Forgive}[above,sloped]{$q$}[below, sloped]{(1,1)}
\istB{Punish}[above,sloped]{$1-q$}[below, sloped]{(-1,-1)}
\endist
\end{istgame}
```

% IGT 218.1 (Osborne, 2004b)

```latex
\begin{istgame}[font=\normalsize]
\xtdistance{10mm}{20mm}
\istroot(0){Vote}
\istb{a}[al]{x}
\istb{b}[ar]
\endist
\istroot(1)([yshift=-1.5em]0-2){Vote}
\istb{c}[al]{y}
\istb{d}[ar]{z}
\endist
\end{istgame}
```
16.2 A game tree with a strategic game

\begin{tabular}{c|c|c|}
\xcol[c]{~} & \xcol[c]{B} & \xcol[c]{S} \\
\hline
B & 3,1 & 0,0 \\
\hline
S & 0,0 & 1,3 \\
\hline
\end{tabular}

\begin{istgame}
\setistmathTF*001\xtdistance{10mm}{40mm} \\
\istroot(0)[initial node]{1} \istb(Book)[al]{2,2} \istb(Concert)[ar] \endist \xtPayoff(0-2){\strategicgame}[below,xshift=-7pt] \end{istgame}

16.3 Larger game trees with information sets

\begin{istgame}
\xtShowEndPoints \xtdistance{10mm}{40mm} \\
\istroot(0)[initial node]{1} \istb \istb \istb \istb \endist \xtdistance{10mm}{10mm} \istroot(1)(0-1) \istb \istb \istb \istb \endist \istroot(2)(0-2) \istb \istb \istb \istb \endist \xtdistance{10mm}{20mm} \istroot(3)(0-3) \istb \istb \istb \endist \xtdistance{10mm}{7mm} \istroot(a)(1-3)[3] \istb \istb \istb \istb \endist \xtdistance{10mm}{14mm} \istroot(b)(2-3) \istb \istb \istb \istb \endist \istroot(c)(3-1)[2] \istb \istb \istb \istb \endist \istroot(d)(3-2)[2] \istb \istb \istb \istb \endist \xtInfosetO[fill=red!20](0)\endist \xtInfosetO[fill=blue!20](a)\endist \setxtinfosetstyle{blue,thick,dashed} \xtInfosetO(b)\endist \setxtinfosetstyle% restore defaults \xtInfosetO(d)\endist \end{istgame}
16.4 A continuum of branches
how-to-visually-present-a-simultaneous-game-with-continuous-strategies

\begin{istgame}[scale=1.5,font=\scriptsize]
\cntmApreset{.9}
\cntmdistance{15mm}{20mm}
\cntrootcntmA(0)[null node]{1}
\istbA(.977)[->,thick]{q_1}[r]
\istbA(0.977)[->,thick]{q_2}[r]
{\vpay{\pi_1(q_1,q_2)\\\pi_2(q_1,q_2)}}
\endist
\end{istgame}

how-to-visually-present-a-simultaneous-game-with-continuous-strategies

\begin{istgame}
\xtdistance{15mm}{40mm}
\cntmdistance{15mm}{30mm}
\cntmApreset[draw=none]{(0)[red!50,opacity=.2]
\cntroot(0){Challenger}
\istb[In]{al} \istb[Out]{ar} \endist
\cntmAistb[draw=none]
\cntrootcntmA(1)(0-1)<135>{Incumbent}
\istbA(q_I)[left] \endist
\cntmAInfosetO(1)
\cntmAistb[draw=none]
\cntrootcntmA(2)(0-2)<45>{Incumbent}
\istbA(q_I)[left]
{\vpay{\pi_C(0,q_I)\\\pi_I(0,q_I)}}
\endist
\cntmApreset[draw=none]{(1)[blue!50,opacity=.2]
\cntmAistb[draw=none]
\cntrootcntmA(3)(1-1)<180>{Challenger}
\istbA*(q_C)[midway, left]
{\vpay{\pi_C(q_C,q_I)\\\pi_I(q_C,q_I)}}
\endist
\end{istgame}
16.5 Tic-tac-toe (sketch)
16.6 Selten’s horse

Selten’s horse: IGT 331.2 (Osborne, 2004b)

\begin{istgame}
\xtdistance{8mm}{16mm}
\istroot[-45]{0} [initial node] {1}
\istb{D}{r}
\istb{grow=east, level distance=30mm} {C}{a}
\endist
\istroot{(0-1)+10mm}{.20mm+}
\istb{L}[a]{1,3,2}
\istb{R}[a]{1,0,0}
\endist
\istroot[-45]{(a)(0-2)}{2}
\istb{d}{r}
\istb{grow=0, level distance=20mm} {c}{a}{1,1,1}[r]
\endist
\istroot{(a1)(a-1)+10mm}{.20mm+}
\istb{L}[a]{4,4,0}
\istb{R}[a]{0,0,1}
\endist
\xtInfoset{(1)(a1)}{3}
\end{istgame}
\captionof{figure}{IGT 331.2}

Figure 2: IGT 331.2

16.7 Centipede game

\begin{istgame}[scale=1.5]
\setistmathTF* 001
\setistgrowdirection{south east}
\xtdistance{10mm}{20mm}
\istroot{(0)} [initial node] {1}
\istb{Take}{r}{(2,0)}[b] \istb{Pass}{a}
\endist
\istroot{(1)}{0-2}{2}
\istb{Take}{r}{(1,3)}[b] \istb{Pass}{a}
\endist
\istroot{(2)}{1-2}{1}
\istb{Take}{r}{(4,2)}[b] \istb{Pass}{a}
\endist
\xtInfoset{(2-2)}{5mm}{2-2}
\xtInfoset{(3)}{5mm}{2-2}
\istroot{(3)}{97,99}{2-2}
\istb{Take}{r}{(97,99)}[b] \istb{Pass}{a}
\endist
\istroot{(4)}{3-2}{1}
\istb{Take}{r}{(100,98)}[b] \istb{Pass}{a}
\endist
\istroot{(5)}{4-2}{2}
\istb{Take}{r}{(99,101)}[b] \istb{Pass}{a}{(100,100)}[r]
\endist
%---------------------
\end{istgame}

% centipede
\begin{istgame}[scale=1.5]
\setistmathTF* 001
\setistgrowdirection{south east}
\xtdistance{10mm}{20mm}
\istroot{(0)} [initial node] {1}
\istb{Take}{r}{(2,0)}[b] \istb{Pass}{a}
\endist
\istroot{(1)}{0-2}{2}
\istb{Take}{r}{(1,3)}[b] \istb{Pass}{a}
\endist
\istroot{(2)}{1-2}{1}
\istb{Take}{r}{(4,2)}[b] \istb{Pass}{a}
\endist
\xtInfoset{(2-2)}{5mm}{2-2}
\xtInfoset{(3)}{5mm}{2-2}
\istroot{(3)}{97,99}{2-2}
\istb{Take}{r}{(97,99)}[b] \istb{Pass}{a}
\endist
\istroot{(4)}{3-2}{1}
\istb{Take}{r}{(100,98)}[b] \istb{Pass}{a}
\endist
\istroot{(5)}{4-2}{2}
\istb{Take}{r}{(99,101)}[b] \istb{Pass}{a}{(100,100)}[r]
\endist
\end{istgame}
16.8 Poker game

% poker: growing south
\begin{lstgame}[scale=1.7]
\setistmathTF*
\xtShowEndPoints
\xtdistance{15mm}{30mm}
\istroot(0)[chance node]{N}
\istB{Black}{al}{\frac12}
\istB{Red}{ar}{\frac12}
\endist
\xtdistance{15mm}{30mm}
\istroot(1-1)(0-1){1}
\istbA(.5)<grow=-135>{Fold}{al}{1,-1}
\istb{Raise}{ar}
\endist
\xtdistance{10mm}{20mm}
\istroot(1)(1-1-2)
\istb{Pass}{al}{1,-1}
\istb{Meet}{ar}{2,-2}
\endist
\xtdistance{15mm}{30mm}
\istroot(1-2)(0-2){1}
\istbA(.5)<grow=-135>{Fold}{al}{-1,1}
\istb{Raise}{ar}
\endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-2-2)
\istb{Pass}{al}{1,-1}
\istb{Meet}{ar}{-2,2}
\endist
\xtInfoset(1)(2){2}
\end{lstgame}
16.9 Poker game: growing to the right

\begin{istgame}[scale=1.3]
\setistmathTF*\setistgrowdirection{0} \xtdistance{15mm}{30mm}
\istroot(0)[chance node]<left>{N}
\istB{Black}[bl]{\frac12}[ar]
\istB{Red}[al]{\frac12}[br]
\endist \xtCInfosetO[fill=blue!20](1-1-2)(1-2-2){2}
\end{istgame}
% poker: growing east -- clockwise (swap version)
\begin{istgame}[scale=1.3]
\setistmathTF*\001
\setistgrowdirection'{0} % \setistgrowdirection'
\xtdistance{15mm}{30mm}
\istroot(0)[chance node]<left>{N}
\istB{Black}[al]{$\frac12$}[br]
\istB{Red}[bl]{$\frac12$}[ar]
\endist
\xtdistance{15mm}{30mm}
\istroot(1-1)(0-1)<left>{1}
\istb{Fold}[al]{1,-1}
\istb{Raise}[bl]
\endist
\xtdistance{12mm}{24mm}
\istroot(1)(1-1-2)
\istb{Pass}[al]{1,-1}
\istb{Meet}[bl]{2,-2}
\endist
\xtdistance{15mm}{30mm}
\istroot(1-2)(0-2)<left>{1}
\istb{Fold}[al]{-1,1}
\istb{Raise}[bl]
\endist
\xtdistance{12mm}{24mm}
\istroot(2)(1-2-2)
\istb{Pass}[al]{1,-1}
\istb{Meet}[bl]{-2,2}
\endist
\xtInfosetO[fill=red!20](1-1-2)(1-2-2){2}
\end{istgame}
16.10 Signaling games

\[ p_1 - E - c_H + p_2^H - c_H, 2H - p_1 - p_2^H \]

\[ p_1 - E - c_H, H - p_1 \]

\[ p_1 - E - c_L + p_2^L - c_L, 2L - p_1 - p_2^L \]

\[ p_1 - E - c_L, L - p_1 \]
\begin{istgame}[scale=1.3]
\xtdistance{20mm}{20mm}
\istroot(0)[chance node]{$c$}
\istb\text{grow=left}{$\frac{1}{2}$}[a]
\istb\text{grow=right}{$\frac{1}{2}$}[a]
\endist
\xtdistance{10mm}{20mm}
\istroot(1)<180>[1]
\istb\text{grow=north}{$a$}[l]
\istb\text{grow=south}{$b$}[l]
\endist
\istroot(2)<0>[1]
\istb\text{grow=north}{$a$}[r]
\istb\text{grow=south}{$b$}[r]
\endist
\istroot'[north](a1)<1-1>
\istb(L)[b1]<-1,0>
\istb(R)[br]<0,-1>
\endist
\istroot(b1)<1-2>
\istb(L)[a1]<2,0>
\istb(R)[ar]<0,2>
\endist
\istroot(a2)<2-2>
\istb(L)[a1]<3,0>
\istb(R)[ar]<0,3>
\endist
\istroot'[north](b2)<2-1>
\istb(L)[b1]<1,0>
\istb(R)[br]<0,1>
\endist
\xtInfoSet(a1)(b2)<2>
\xtInfoSet(b1)(a2)<2>
\end{istgame}
17 Selected examples from tex.stackexchange.com

In this section \setistgamefontsize{\normalsize}.

17.1 Subgames: \xtSubgameBox

\begin{game}
\xtShowEndPoints
\xtDistance{25mm}{70mm}
\istrooto{0}{Firm 1}
\istb{BIG}{al} \istb{small}{ar} \endist
\xtDistance{25mm}{35mm}
\istrooto{1}{0-1}{Firm 2}
\istb{Low}{al} \istb{High}{ar} \endist
\istrooto{2}{0-2}{Firm 2} \istb{low}{al} \istb{high}{ar} \endist
\xtDistance{20mm}{10mm}
\istrooto{a}{1-1}{Firm 1}
\istb{L}{lc}\parbox{1em}{$u_1$
\parbox{1em}{$u_2$}} \istb{H}{rc} \endist
\istrooto{b}{1-2}{Firm 1}
\istb{L}{lc} \istb{H}{rc} \endist
\istrooto{c}{2-1}{Firm 1}
\istb{l}{lc} \istb{h}{rc} \endist
\istrooto{d}{2-2}{Firm 1}
\istb{l}{lc} \istb{h}{rc} \endist
\setxtinfosetstyle{dashed}
\xtInfoset(a){b}
\xtInfoset(c){d}
\xtSubgameBox{yshift=-2ex}{(a-1)(b-2)}{inner ysep=7ex}
\xtSubgameBox{yshift=-2ex}{(c-1)(d-2)}{inner ysep=7ex}
\end{game}
17.2 Information sets: $\text{\xtInfosetO}$, $\text{\xtCInfosetO}$, and $\text{\cntmAInfosetO}$
\begin{istgame}[scale=.55, font=\footnotesize]
\setistmathTF*001
\% top part
\xtdistance{25mm}{120mm}
\istroot(0)[chance node]{Nature} \istb \indist
\xtdistance{25mm}{50mm}
\istroot(A)(0-1) \istb \indist
\istroot(B)(0-2) \istb \indist
\% left part
\xtdistance{25mm}{60mm}
\istroot(A0)(A-1)<180>{2} \istb \istb \indist
\xtdistance{25mm}{30mm}
\istroot(A1)(A0-1) \istb \istb \indist
\istroot(A2)(A0-2) \istb \istb \indist
\xtdistance{25mm}{15mm}
\istroot(A3)(A1-1)<180>{2} \istb[G][1] \istb[S][r] \indist
\istroot(A4)(A1-2)<180>{2} \istb[G][1] \istb[S][r] \indist
\istroot(A5)(A2-1)<180>{2} \istb[G][1] \istb[S][r] \indist
\istroot(A6)(A2-2)<180>{2} \istb[G][1] \istb[S][r] \indist
\% right part
\xtdistance{25mm}{60mm}
\istroot(B0)(B-1)<180>{2} \istb \istb \indist
\xtdistance{25mm}{30mm}
\istroot(B1)(B0-1) \istb \istb \indist
\istroot(B2)(B0-2) \istb \istb \indist
\xtdistance{25mm}{15mm}
\istroot(B3)(B1-1)<180>{2} \istb[G][1] \istb[S][r] \indist
\istroot(B4)(B1-2)<180>{2} \istb[G][1] \istb[S][r] \indist
\istroot(B5)(B2-1)<180>{2} \istb[G][1] \istb[S][r] \indist
\istroot(B6)(B2-2)<180>{2} \istb[G][1] \istb[S][r] \indist
\% information sets
\setxtinfosetstyle{dashed}
\xt InfosetO[fill=black!15]{(A)(B){1}}
\xt InfosetO[blue, fill=blue!15]{(A1)!1.15}(B1)<.35>{1}
\xt InfosetO[red, fill=red!15]{(A2)!0.75}(B2)<.75>{1}
\end{istgame}
how-to-design-a-game-in-a-tree
\begin{istgame}[scale=1.5]
\setistgrowdirection'{east}

%% root
\xtdistance{20mm}{32mm}
\istroot(0)[chance node]{0}
\istb \istb \istb \istb \endist

%% extending branches
\istroot(0a)(0-1)[null node]
\istb{\frac{1}{3}(TL)}[above,near start] \endist
\istroot(0b)(0-2)[null node]
\istb{\frac{1}{3}(BL)}[above,near start] \endist
\istroot(0c)(0-3)[null node]
\istb{\frac{1}{3}(TR)}[below,near start] \endist
\istroot(0d)(0-4)[null node]
\istb{0(BR)}[below,near start] \endist

%\xtShowEndPoints

%% player I
\xtdistance{15mm}{16mm}
\istroot(1a)(0a-1)
\istb{T_1}[a] \istb{B_1}[b] \endist
\istroot(1b)(0b-1)
\istb{T_2}[a] \istb{B_2}[b] \endist
\istroot(1c)(0c-1)
\istb{T_1}[a] \istb{B_1}[b] \endist
\istroot(1d)(0d-1)
\istb{T_2}[a] \istb{B_2}[b] \endist

\xtdistance{15mm}{8mm}

%% player II
\istroot(2Aa)(1a-1)
\istb{L_1}[a]{6,6} \istb{R_1}[b]{2,7} \endist
\istroot(2Ab)(1a-2)
\istb{L_1}[a]{7,2} \istb{R_1}[b]{0,0} \endist
\istroot(2Ac)(1b-1)
\istb{L_1}[a]{6,6} \istb{R_1}[b]{2,7} \endist
\istroot(2Ad)(1b-2)
\istb{L_1}[a]{7,2} \istb{R_1}[b]{0,0} \endist
\istroot(2Ba)(1c-1)
\istb{L_2}[a]{6,6} \istb{R_2}[b]{2,7} \endist
\istroot(2Bb)(1c-2)
\istb{L_2}[a]{7,2} \istb{R_2}[b]{0,0} \endist
\istroot(2Bc)(1d-1)
\istb{L_2}[a]{6,6} \istb{R_2}[b]{2,7} \endist
\istroot(2Bd)(1d-2)
\istb{L_2}[a]{7,2} \istb{R_2}[b]{0,0} \endist

% information sets
\xtInfosetO(0)
\xtCInfosetO[fill=red!20,fill opacity=.3](1a)!.4!(1c)<1.2>{II}
\xtCInfosetO[fill=blue!20,fill opacity=.3](1b)!.6!(1d)<1.15>{II}
\xtInfosetO(2Aa)(2Ad){II}
\xtInfosetO(2Ba)(2Bd){II}

\end{istgame}
17.3 Continuum of moves: \texttt{\textbackslash istrootcntm} and \texttt{\textbackslash istrootcntmA}

\texttt{\textbackslash draw-tree-in-tikz}

\begin{tikzpicture}
\tikzset{odd node/.style={decision node,minimum size=6pt}}
\tikzset{even node/.style={oval node,fill=cyan!50,minimum size=6pt}}
\% game tree
\istgame
% presets
\cntmpreset*[densely dashed]{.6}
\istrootcntm(1a)[odd node]<15>(\$x_1$)
\istb{(a^0,1-a^0)}[right,near end] \endist
\istrootcntm(2)(1a-1)[even node]<-90>(\$x_2$)
\istb{acc}[al]{(a^0,1-a^0)}
\istb{not}[ar]
\endist
\istrootcntm(3a)(3-2)[odd node]<0>(\$x_3$)
\istb{(a^1,1-a^1)}[right,near end] \endist
\istrootcntm(4)(3a-1)[even node]<-90>(\$x_4$)
\istb{acc}[al]{(\delta_1 a^1,\delta_2(1-a^1))}
\istb{not}[ar]{(0,0)}
\endist
\istrootcntm(5a)(5-2)[odd node]<0>(\$x_5$)
\istb{(a^2,1-a^2)}[right,near end] \endist
\istrootcntm(6)(5a-1)[even node]<-90>(\$x_6$)
\istb{acc}[al]{((\delta_1)^2a^2,((\delta_2)^2(1-a^2))}
\istb{not}[ar]{(0,0)}
\endist
\end{istgame}

\end{tikzpicture}
\begin{istgame}[scale=1.3]
% nodes
\tikzset{p1/.style={oval node,minimum size=3pt}}
\tikzset{p2/.style={oval node,fill=black,minimum size=3pt}}
% tree
\cntmdistance{10mm}{20mm}
\istrootcntmA(2a)(1-1)[p2]\istbA[draw=none]\endist
\istrootcntmA(2b)(1-2)[p2]\istbA[draw=none]\endist
\xtdistance{15mm}{30mm}
\istroot(1a)(2a-1)[p1]\istb\istb\endist
\istroot(1b)(2b-1)[p1]\istb\endist
\istrootcntmA(2c)(1a-1)[p2]\istbA[draw=none]\endist
\istrootcntmA(2d)(1a-2)[p2]\istbA[draw=none]\endist
\istrootcntmA(2e)(1b-1)[p2]\istbA[draw=none]\endist
\istroot(1c)(2c-1)[p1]\istb\endist
\istroot(1d)(2d-1)[p1]\istb\endist
\istroot(1e)(2e-1)[p1]\istb\endist
% information sets
\xtInfosetO[solid,thin,fill=blue!10](2a)(2b)
\xtInfosetO[solid,thin,fill=blue!10](2d)(2e)
\end{istgame}
a-gametree-with-variable-choices

\begin{istgame}[scale=1.3,semithick]
\tikzset{oval node/.style={box node,draw=none,minimum size=5mm}}
\cntmdistance*{20mm}{25mm}
\istrooto{0}{plain node}{Firm 1}+15mm..50mm+
\setistmathTF
\cntmApreset{.6}
\istrootocntmA{(E2)}{(0-1)}{E_2} \istb[thin] \endist
\xtNode*{(cntm-1)}{0} \xtNode*{(cntm-2)}{\infty}
\istrootocntmA{(E3)}{(0-2)}{E_3} \istb[thin] \endist
\xtNode*{(cntm-1)}{0} \xtNode*{(cntm-2)}{\infty}
\istrootocntmA{(11)}{(E2-1)}{q_1} \istb[thin] \endist
\xtNode*{(cntm-1)}{0} \xtNode*{(cntm-2)}{\infty}
\istrootocntmA{(21)}{(E3-1)}{q_1} \istb[thin] \endist
\xtNode*{(cntm-1)}{0} \xtNode*{(cntm-2)}{\infty}
\istrootocntmA{(2a)}{(11-1)}{q_2} \istb[thin] \endist
\xtNode*{(3a)}{(11-1)}{(c_2)^{c_2}\prod_2^c\prod_3^m}
\istrootocntmA{(3b)}{(21-1)}{q_3} \istb[thin] \endist
\xtNode*{(3a)}{(21-1)}{(c_2)^{c_2}\prod_2^c\prod_3^m}
\istrootocntmA{(3b)}{(21-1)}{(c_3)^{c_3}\prod_2^m\prod_3^c}
\end{istgame}
17.4 Signaling games

`how-draw-this-game-in-tikz`

\begin{tikzpicture}
\setistNewNodeStyle{solid node}[null node]
\setistmathTF
\setxtshowarrows[thick]
\xtdistance
% game start
\setistgrowdirection'{east}
\istroot(0)[chance node]<[xshift=2mm]200>{Nature}
\istb{[.5]1} \istb{[.5]2} \endist
% right part
\istrootRa(0-1)<45>{\$x_1\$} \istb{Average}[a] \endist
\istrootRb(0-2)<45>{\$x_2\$} \istb{Average}[b] \endist
\istroot{20mm}{20mm}
\istroot{T1a}(Ra-1)<135>{\$x_3\$} \istb{Hunk}[a] \istb{Average}[below,sloped]{2,2} \endist
\istroot{T1b}(Rb-1)<45>{\$x_4\$} \istb{Hunk}[b] \istb{Average}[below,sloped]{2,2} \endist
\xtInfoset[dashed](0-1)(0-2){h_{T1}} \xtInfosetOwner(T1a)(T1b){Tina}[l]
\xtInfoset[dashed](T2a)(T2b){h_{T2}}[r]
\xtInfosetOwner(T2a)(T2b){Tina}[l]
% left part
\setistgrowdirection{west}
\istrootLa(0-1) \istb{Hunk}[a] \endist
\istrootLb(0-2) \istb{Hunk}[b] \endist
\istroot{20mm}{20mm}
\istroot{T2a}(La-1)<-45>{\$x_5\$} \istb{Hunk}[a] \istb{Average}[below,sloped]{3,2} \endist
\istroot{T2b}(Lb-1)<45>{\$x_6\$} \istb{Hunk}[b] \istb{Average}[below,sloped]{3,2} \endist
\xtInfoset[dashed](T2a)(T2b){h_{T1}}[r]
\xtInfosetOwner(T2a)(T2b){Tina}[l]
\end{tikzpicture}
17.5 Cross out nodes

problem-with-dashed-edges-and-cross-nodes
17.6 Large game trees
\begin{istgame}[scale=.8,font=\footnotesize]
\xtShowEndPoints % solid nodes
\setistEllipseNodeStyle(6mm) % minimum circle size for players
\xtdistance(30mm){30mm}
\istrooto(0){Nature}
\iset{JKQ}{1}[l]{p_1}[left,near start,xshift=-5pt]
\iset{JQK}{1}[l]{p}[left,near start]
\iset{QJK}{1}[r]{p}[right,near start]
\iset{KJQ}{1}[r]{p}[right,near start,xshift=5pt]
\iset{KQJ}{1}[r]{p}[right,near start,xshift=5pt]
\endist
\xtdistance(25mm){10mm}
\istrooto(1){0-1}\set{SP}{18}
\isetb{c}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{-1}{1}
\istrooto(2){0-2}\set{SP}{18}
\isetb{c}{1}\set{\vpay}{1}{1}\isetb{r}{r}\set{\vpay}{-1}{1}
\istrooto(3){0-3}\set{SP}{18}
\isetb{r}{1}\isetb{c}{r}\set{\vpay}{1}{1}\isetb{r}{r}\set{\vpay}{1}{1}
\istrooto(4){0-4}\set{SP}{18}
\isetb{c}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{-1}{1}
\istrooto(5){0-5}\set{SP}{18}
\isetb{c}{1}\set{\vpay}{1}{1}\isetb{r}{r}\set{\vpay}{1}{1}
\istrooto(6){0-6}\set{SP}{18}
\isetb{c}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{1}{1}
\istrooto(1){2-2}\set{SP}{28}
\isetb{c}{1}\set{\vpay}{1}{1}\isetb{r}{r}\set{\vpay}{1}{1}
\istrooto(2){3-1}\set{SP}{28}
\isetb{f}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{1}{1}
\istrooto(3){4-2}\set{SP}{28}
\isetb{f}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{1}{1}
\istrooto(4){5-1}\set{SP}{28}
\isetb{f}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{1}{1}
\istrooto(5){6-1}\set{SP}{28}
\isetb{f}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{1}{1}
\xtdistance(28mm){25mm}
\istrooto(A){1-2}\set{SP}{28}
\isetb{f}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{1}{1}
\istrooto(B){6-1}\set{SP}{28}
\isetb{f}{1}\set{\vpay}{1}{1}\isetb{c}{r}\set{\vpay}{1}{1}
\endist
% information sets
\setxtinfosetstyle{dashed}
\xtInfoset(1){2}
\xtInfoset(3){4}
\xtInfoset(5){6}
\xtInfoset(a){b}
\xtInfoset(c){d}
\xtInfoset(A){B}
\end{istgame}
17.7 Probability trees

\begin{tikzpicture}[scale=0.7]
% TikZ code goes here...
\end{tikzpicture}
17.8 More tree examples

game-tree-in-microeconomics

\begin{tikzpicture}
\tikzset
\node (root) at (0,0) {\{a,b\}};
\node (a) at (-1,-1) {a};
\node (b) at (1,-1) {b};
\node (u-l) at (-2,-2) {U,L};
\node (u-r) at (2,-2) {U,R};
\node (d-l) at (-2,-3) {D,L};
\node (d-r) at (2,-3) {D,R};
\node (P) at (4,-1) {P};
\node (N) at (-4,-1) {N};
\draw[->] (root) -- (a);
\draw[->] (root) -- (b);
\draw[->] (root) -- (u-l);
\draw[->] (root) -- (u-r);
\draw[->] (root) -- (d-l);
\draw[->] (root) -- (d-r);
\end{tikzpicture}

constructing-a-tree-in-latex-using-tikz

\begin{tikzpicture}
\node (A) at (0,0) {A};
\node (B) at (-1,-1) {B};
\node (C) at (-2,-2) {C};
\node (D) at (1,-2) {D};
\node (E) at (2,-1) {E};
\draw (A) -- (B);
\draw (A) -- (C);
\draw (A) -- (D);
\draw (A) -- (E);
\end{tikzpicture}
\begin{istgame}[scale=1.5,draw=blue,text=blue,font=\sffamily]
\tikzset{every node/.style={fill=white}} % action labels with white background
\tikzset{every ellipse node/.style={circle,draw=blue,minimum size=2.5em}}
\tikzset{xx/.style={circle,draw,fill,minimum size=2.5em,text=black}} % color node
\tikzset{lev/.style=(level distance=#1)}
\setistmathTF*000{\sffamily}
\setistgrowdirection'{east}
\xtShowEndPoints[oval node]
\def\xdist{50mm}
\% root
\istrooto(A){Root}
\istb<grow=0,lev=.4*\xdist>{1}{\istb<grow=-90,lev=.7*\xdist>{0}{pos=.7}}
\dist\istdistance{.4*\xdist}{.4*\xdist}
\istrooto(A-1){A\(A-1\)}\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb\istb
Retailer announces $P$ and $T$

Retailer

Consumer

Consumer

Stage I

Stage II

Stage III

Utility

Match

Mismatch

Buy

Don’t buy

Keep

Return

Keep

Return

$v - P$

$vc - h$

$s - P$

$-h$

$0$

$\begin{istgame}\text{\texttt{edge from parent path=\{}}\begin{tikzparentnode} |- (\tikzchildnode.west)\text{\}\}}\end{istgame}$

\begin{tikzpicture}

\setistgrowdirection '{east}'
\setistmathTF 001
\setxtarrowtips{latex}{very thick}
\setistNewNodeStyle {init}[rectangle,fill=gray]{1cm}
\setistNewNodeStyle {rect}[rectangle]{1cm}
\setistNewNodeStyle {circ}{1cm}
\tikzset
\RR/.style={edge from parent path={(\tikzparentnode.east)--(\tikzchildnode.west)}}
\% tree
\istroot (0)[init]<[xshift=3mm]-90>{\makecell[l]{Retailer\announces\$P$ and $T$}}
\istroot (1)[(0-1)[rect]
\istb [-r-.75] \endist % \usepackage{makecell}
\istroot (1)[(0-1)[rect]
\istb <RR>[->-=.75]{Buy}[above,pos=.3]
\istb <sibling distance=60mm,level distance=65mm>{Don’t buy}[above,pos=.57]{0}
\endist
\istroot (2)[(1-1)[circ]
\istB [r-.93]{Match}[above,pos=.72]{\$m\$}[below,pos=.72]
\istB [r-.93]{Mismatch}[below,pos=.72]{\$1-m\$}[above,pos=.72] \endist
\istroot (3a)[(2-1)[rect]
\istb[{v-P}]{Keep}[above,pos=.75]{v-P}
\istb{Return}[below,pos=.75]{vc-h} \endist
\istroot (3b)[(2-2)[rect]
\istb[{s-P}]{Keep}[above,pos=.75]{s-P}
\istb{Return}[below,pos=.75]{-h} \endist
\% time-lines
\xtTimeLineV[dashed]{(\texttt{[xshift=-7mm]1})\{3.5\}{-4}{Stage I}[left=5mm]
\xtTimeLineV[dashed]{(\texttt{[xshift=-7mm]2})\{3.5\}{-4}{Stage II}[left=5mm]
\xtTimeLineV[dashed]{(\texttt{[xshift=-1.5mm]3a-1})\{3.5\}{-4}{Stage III}[left=15mm]
\xtTimeLineV[draw=none]{(\texttt{[xshift=-1.5mm]3a-1})\{3.5\}{-4}{\underline{Utility}}}
\xtTimeLineV[draw=none]{(\texttt{[xshift=-7mm]1})\{3.5\}{-4}{Retailer}[left=5mm]
\xtTimeLineV[draw=none]{(\texttt{[xshift=-7mm]2})\{3.5\}{-4}{Consumer}[left=5mm]
\xtTimeLineV[draw=none]{(\texttt{[xshift=-1.5mm]3a-1})\{3.5\}{-4}{Consumer}[left=15mm]
\end{istgame}
\begin{istgame}
\setistNewNodeStyle{max}
  [regular polygon, regular polygon sides = 3]{1.5cm}
\setistNewNodeStyle{min}
  [regular polygon, regular polygon sides = 3, shape border rotate = 180]{1.5cm}
\setistNewNodeStyle{chance}
  [circle]{1.2cm}
\def\distFactor{20};
\xtdistance{\distFactor mm}{4*\distFactor mm}
\setxtarrowtips[blue, thick]
\istroot{(0)}[max]<center, blue>{1.5}
  \istb{0.5}[al]
  \istb{0.5}[ar]
\endist
\xtdistance{\distFactor mm}{2*\distFactor mm}
\istroot{(1)}{0-1}[chance]<center, purple>{1.5}
  \istb{0.5}[al]
  \istb{0.5}[ar]
\endist
\istroot{(2)}{0-2}[chance]<center, purple>{$\leq 1$}
  \istb{0.5}[al]
  \istb{0.5}[ar]
\endist
%% terminal nodes with or without circles (TRICK!!)
\xtShowEndPoints[circle,draw,minimum size=1.1cm] % before \xtShowTerminalNodes
\xtShowTerminalNodes[circle,draw=none,minimum size=1.1cm]
\xtdistance{\distFactor mm}{\distFactor mm}
\istroot{(3)}{1-1}[min]<center, red>{2}
  \istbt{}{1}[center]
  \istb{}{5000}[center]
\endist
\istroot{(4)}{1-2}[min]<center, red>{1}
  \istbt{}{1}[center]
  \istb{}{100}[center]
\endist
\istroot{(5)}{2-1}[min]<center, red>{0}
  \istb{}{0}[center]
\endist
\istroot{(6)}{2-2}[min]
  \istb{}{-1}[center]
\endist
\end{istgame}
Deviate or Collude

Deviate

\( R_{dev} - EDC \)

\( R_{comp} * \delta \)

\( R_{comp} * \delta^2 \)

\( R_{comp} * \delta^3 \)

... 

Collude

\( R_{cartel} - EDC - F \)

\( R_{comp} * \delta \)

\( R_{comp} * \delta^2 \)

\( R_{comp} * \delta^3 \)

... 

\( R_{cartel} \)

\( R_{cartel} * \delta \)

\( R_{cartel} * \delta^2 \)

\( R_{cartel} * \delta^3 \)

... 

\( 1 - \alpha \)

\( 1 - \alpha \)

\( 1 - \alpha \)

\( 1 - \alpha \)
\begin{tikzpicture}
\def\Rcar{{R_{cartel}}}
\def\Rcom{{R_{comp}}}
% tree
\tikzset{oval node/.style={box node,draw=none,outer sep=1pt}}
\xtdistance{15mm}{50mm}
\istrooto{(0)}{Deviate or Collude} \istb \istb \endist
\istrooto{(Da)}{(D-1)}{$R_{\text{dev}}-\text{EDC}$} \istb \endist
\istrooto{(Db)}{(Da-1)}{$\Rcom*\delta$} \istb \endist
\istrooto{(Dc)}{(Db-1)}{$\Rcom*\delta^2$} \istb \endist
\istrooto{(Dd)}{(Dc-1)}{$\Rcom*\delta^3$} \istb{\cdots} \endist
\xtdistance{15mm}{50mm}
\istrooto{(C)}{Collude} \istb{\alpha} \istb{1-\alpha} \endist
\istrooto{(Ca)}{(C-1)}{($R_{\text{cartel}}$)} \istb \endist
\istrooto{(Cb)}{(Ca-1)}{$\Rcom*\delta^2$} \istb \endist
\istrooto{(Cc)}{(Cb-1)}{$\Rcom*\delta^3$} \istb{\cdots} \endist
\xtdistance{15mm}{40mm}
\istrooto{(RC0)}{($R_{\text{cartel}}$)} \istb{\alpha} \istb{1-\alpha} \endist
\istrooto{(RC0a)}{(RC0-1)}{($R_{\text{cartel}}$)} \istb{\cdots} \endist
\istrooto{(RC0b)}{(RC0a-1)}{$\Rcom*\delta^2$} \istb \endist
\istrooto{(RC0c)}{(RC0b-1)}{$\Rcom*\delta^3$} \istb{\cdots} \endist
\xtdistance{15mm}{30mm}
\istrooto{(RC1)}{($R_{\text{cartel}}$)} \istb{\alpha} \istb{1-\alpha} \endist
\istrooto{(RC1a)}{(RC1-1)}{($R_{\text{cartel}}$)} \istb{\cdots} \endist
\istrooto{(RC1b)}{(RC1a-1)}{$\Rcom*\delta^3$} \istb{\cdots} \endist
\xtdistance{15mm}{20mm}
\istrooto{(RC2)}{($R_{\text{cartel}}$)} \istb{\alpha} \istb{1-\alpha} \endist
\istrooto{(RC2a)}{(RC2-1)}{($R_{\text{cartel}}$)} \istb{\cdots} \endist
\istrooto{(RC2b)}{(RC2a-1)}{$\Rcom*\delta^3$} \istb \endist
\end{tikzpicture}
Version history

- v2.1 (2022/03/22) uploaded to CTAN
  - document done!
- v2.092 (2022/03/21)
  - working on the documentation
  - added arrows.meta and bending to the list of preloaded tikz libraries
    * applied arrows.meta arrow tips to the related examples
- v2.091 (2022/03/16)
  - modified istgame environment to add its starred version
    * \begin{istgame} is essentially the same as \begin{tikzpicture}
    * but loses some feature of the istgame package
  - fixed all macros related information sets to remove arrow tips
  - touched \AtBeginDocument{\setxtarrowtips}
  - rewrote \setistmathTF(*) and related macros to avoid a possible conflict
- v2.09 (2021/04/05)
  - removed \pgfextra from \lstroottcntm and \lstroottcntmA (debugging)
  - resolved the null font problem in \cntmistb and \cntmAistb using \l_nullfont_tl (debugging)
  - added \setist<...>nodesize for solid, hollow, ellipse, and rectangle (for global setting in the preamble)
- v2.0 (2019/01/27) uploaded to CTAN
  - changed the title of the package to “Draw Game Trees with Ti\kZ”
  - package document done
- v2.0rc (2019/01/27)
  - introduced \setxtinfosetstyle
    * introduced infoset style, a new style for information sets
    * modified all definitions of information sets, accordingly
  - redefined macros related to layer, to easily restore default values
  - modified the cross arrow tip options
  - modified defaults of \setxtarrowtips not to depend on branch styles initially
  - code refinement
- v2.0beta1 (2019/01/26)
  - introduced \xtshowXpoints and \xtshowXcircles (for developer only, not documented)
- v1.0 (2017/09/04)
  - introduced \cntmlevdist and \cntmsibdist
  - introduced \cntmdistance
  - redefined related macros
  - package document done

129
Acknowledgement

The basic idea of the istgame package came from Osborne’s egameps package and Chen (2013). The update of the package istgame to version 2.0 is partly motivated by the questions and discussions on the game tree topic at https://tex.stackexchange.com/. Special thanks go to Kangsoo Kim of KTUG for his great help in using expl3 to simplify the appearance and the usage of the macros, and to resolve the issues on TeX’s expansion.

References

Index

【 Symbols 】
->- 78
->>>- 78
-o- 78
-x- 78

【 A 】
abbreviations 22, 38, 43, 44, 85
action 9, 21, 33
action sibling distance 61, 62
arrow 77
asymmetric scaling 59

【 B 】
backward induction 95
box node 14
branch 9, 21, 33

【 C 】
chance node 14
circle arrow tip 78
clockwise 18, 20, 46, 85
\cntmAexpostShowEndPoints 71
\cntmAInfoset 73
\cntmAInfosetO 74
\cntmAistb 71
\cntmAistb* 71
\cntmAlayerpreset 69
\cntmApreset 67, 68
\cntmdistance 62
\cntmdistance* 12, 63
\cntmistb 70
\cntmistb* 70
\cntmlevdist 60
\cntmpreset 63
\cntmpreset* 64
\cntmsibdist 60
continuum 11, 60
counterclockwise 17, 20, 46, 85
cross arrow tip 78
curved information set 54

directional 8

effective distance 8

【 D 】
decision nodes 14
direction 11, 37, 45
double arrow tip 78
dual action labels 27, 35

effective distance 8

【 E 】
ellipse node 13
\endist 8
exp3 8

【 H 】
hollow node 13

【 I 】
information set 10, 49, 50, 54, 56, 73, 74
info set style 49
initial node 14
\istB 27, 35
\istB 8, 9, 21, 33, 36
\istB* 27
\istB* 25
\istBA 29, 66
\istBAt 29
\istBAt* 29
\istbm 22
\istBt 35
\istBtt 26
\istBt* 27
\istBtt* 27
\istgame i, 8, 13, 59
\istgrowdirection 17, 24, 37
\istroot' 18
\istroot 8, 9, 16, 30
\istrootcntm' 62
\istrootcntm 11, 60
\istrootcntmA' 66
\istrootcntmA 65, 66
\istrooto' 20
\istrooto 19, 31
\istrootocntm' 69
\istrootocntm 69
\istrootocntmA' 69
\istrootocntmA 69

【 L 】
layer 53, 69, 91
level distance 8

【 M 】
middle arrow 78
middle arrow tip style 78
midpoint factor 54, 55, 57
missing 22

【 N 】
null node 13

【 O 】
oveal node 14
owner 9, 16, 30

extensive tree 12

131
【P】
payoff 9, 21
plain node 13, 84
player 9, 16, 30
plot factor 54, 55, 57, 73

【R】
rectangle node 13
root 9, 14, 16

【S】
scale 59
\setist<...>NodeStyle 6, 14
\setistdefaultnodedrawcolor 6
\setistdefaultnodefillcolor 6
\setistdefaultnodeinnersep 6
\setistdefaultnodeoutersep 6
\setistellipsenodesize 2
\setistgamefontsize 6, 13
\setistgameshorten 6, 77
\setistgrowdirection' 6, 21, 45, 47, 85
\setistgrowdirection 6, 20, 36, 45
\setisthollownodesize 2
\setistmathTF 6, 35, 39
\setistmathTF* 6, 36, 40, 41
\setistNewNodeStyle 14
\setistrectanglenodesize 2
\setistsolidnodesize 2
\setxtarrowtips 79
\setxtgrowkey 48
\setxtinfosetlayer 53, 58
\setxtinfosetstyle 52
\setxtshowarrows 82
\setxtshowmidarrows 80
\setxtsubgamelen 91
shrink factor 64
sibling distance 8
simple tree 8
simple triangle 64
single arrow tip 78
singleton information set 50, 56
sloped information set 52
sloped labels 22, 34
solid node 13
square node 14
subgame 88

【T】
terminal nodes 14
tikz 8
\tikzpicture 13
transform shape 15
triple arrow tip 78

【X】
\xparse 8
\xtscale 59
\xtActionLabel 28, 41, 83
\xtActionLabel* 41, 83
\xtALPush 43
\xtALShift 45
\xtCInfoset 54
\xtCInfoset* 54
\xtCInfosetO 56, 57
\xtCInfosetO* 56
\xtCInfosetOTurnX 58
\xtCommentFrom 87
\xtCommentTo 87
\txcureslopedlabelsEW 23
\txcureslopedlabelsNS 23
\txdistance 12
\txHideArrows 91
\txHideArrows* 91
\txHideEndPoints 26
\txHideEndPoints* 70, 71
\txHideMidArrows 79
\txHideTerminalNodes 26
\txInfo 10, 49
\txInfo* 49
\txInfoO 50
\txInfoO* 50
\txInfoOwner 41, 84
\txInfoOwner* 84
\txlevdist 12
\txNode 84
\txNode* 84
\txPayoff 41, 83
\txPayoff* 84
\txscale 13
\txShowArrows 81
\txShowEndPoints 25
\txShowEndPoints* 26, 72
\txShowMidArrows 79
\txShowTerminalNodes 26
\txsibdist 12
\txSubgameBox 88
\txSubgameBox* 89
\txSubgameOval 89
\txSubgameOval* 89
\txTimeLineH' 85
\txTimeLineH 85
\txTimeLineV' 86
\txTimeLineV 86
\txtxscale 13
\txtyscale 13

【Y】
\yscale 59