Abstract

This is a LaTeX package that provides macros based on TikZ to draw a game tree. 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 istgame package, you can draw a game tree as if you were drawing a game tree with a pen and paper.

KEYWORDS: game trees, nodes, branches, information sets, subgames
6.2.2 Changing the color of action labels and the style of branches ........................................... 16
6.3 How to put payoffs ......................................................................................................................... 16
   6.3.1 Payoffs: basics ......................................................................................................................... 16
   6.3.2 Payoffs and south .................................................................................................................... 16
   6.3.3 Coloring payoffs ...................................................................................................................... 18
7 Fine-tuning positions of players, action labels, and payoffs (experimental) .................................. 18
   7.1 Fine-tuning positions: owners ..................................................................................................... 18
   7.2 Fine-tuning positions: action labels .......................................................................................... 18
       7.2.1 Abbreviations: [l], [c], [a], and [b] .................................................................................... 18
       7.2.2 Abbreviations: [al], [ar], [bl], and [br] ............................................................................ 19
   7.3 Fine-tuning positions: payoffs ................................................................................................... 21
8 Growing direction of trees ................................................................................................................. 21
   8.1 \setistgrowdirection \text{–} counterclockwise .............................................................................. 21
   8.2 \setistgrowdirection \text{–} clockwise ............................................................................................. 22
   8.3 Examples of rotating trees with \setistgrowdirection(*) ............................................................... 22
       8.3.1 A tree growing east \text{–} counterclockwise ........................................................................ 22
       8.3.2 A tree growing east \text{–} clockwise .................................................................................. 23
       8.3.3 A tree growing north \text{–} counterclockwise ....................................................................... 23
       8.3.4 A tree growing north \text{–} clockwise ................................................................................ 24
       8.3.5 \setistgrowkey for one simple tree .................................................................................. 24
9 Information sets .................................................................................................................................. 25
   9.1 Information sets: \XtInfoset(*) ..................................................................................................... 25
   9.2 Information sets: \XtInfoset0(*) (experimental) .......................................................................... 26
10 Continuum of branches ..................................................................................................................... 27
   10.1 \XtCntm (standard version) .......................................................................................................... 27
       10.1.1 Basics .................................................................................................................................. 27
       10.1.2 Changing the size and the color of a continuum of branches ............................................. 28
   10.2 \XtCntmarc (arc version) ............................................................................................................ 29
       10.3 Examples ............................................................................................................................... 29
11 Auxiliary macros ............................................................................................................................... 31
12 Representing subgames .................................................................................................................... 33
   12.1 \XtSubgameBox(*) (experimental) ............................................................................................... 33
       12.1.1 \XtSubgameBox .................................................................................................................... 33
       12.1.2 \XtSubgameBox* .................................................................................................................. 34
   12.2 \XtSubgameOrt1(*) (experimental) ............................................................................................. 34
   12.3 More examples .......................................................................................................................... 36
13 Miscellany .......................................................................................................................................... 37
   13.1 Various branch types, directions, and lengths ............................................................................ 37
   13.2 Code reuse ................................................................................................................................... 37
       13.2.1 Drawing subgames .............................................................................................................. 37
       13.2.2 Backward induction .......................................................................................................... 39
       13.2.3 Code reusability ................................................................................................................. 40
14 Game tree examples .......................................................................................................................... 41
   14.1 Simple examples .......................................................................................................................... 41
   14.2 A game tree with a strategic game ............................................................................................. 41
   14.3 Larger game trees ....................................................................................................................... 43
   14.4 Tic-tac-toe (sketch) .................................................................................................................... 44
   14.5 Selten’s horse ............................................................................................................................. 45
   14.6 Centipede game ......................................................................................................................... 45
   14.7 Poker game .................................................................................................................................. 46
   14.8 Poker game: growing to the right ............................................................................................... 47
   14.9 Signaling games .......................................................................................................................... 49

Version history .......................... 51
Acknowledgement ........................ 52
References .................................. 52
Index ........................................ 53
0 Changes

A considerable number of macro names have been changed in the version 0.8 (Jan. 17, 2017) of this package. The following old macro names in any previously written documents using codes in istgame ver. 0.7 or before, should be replaced by the new names, accordingly.

Also, \istroot*, \istcntm*, and \xtInfoset* should be replaced by \istrooto, \istcntmarc, and \xtInfoseto, respectively, in the version 1.0 or later.

<table>
<thead>
<tr>
<th>Old Macro Name</th>
<th>New Macro Name</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>\xdistance</td>
<td>\xtdistance</td>
<td>0.8</td>
</tr>
<tr>
<td>\xDot</td>
<td>\xtNode</td>
<td>0.8</td>
</tr>
<tr>
<td>\xInfoset</td>
<td>\xtInfoset</td>
<td>0.8</td>
</tr>
<tr>
<td>\xInfoset*</td>
<td>\xtInfoset*</td>
<td>0.8</td>
</tr>
<tr>
<td>\xInfosetOwner</td>
<td>\xtInfosetOwner</td>
<td>0.8</td>
</tr>
<tr>
<td>\xAActionLabel</td>
<td>\xtActionLabel</td>
<td>0.8</td>
</tr>
<tr>
<td>\xPayoff</td>
<td>\xtPayoff</td>
<td>0.8</td>
</tr>
<tr>
<td>\ShowTerminalNodes</td>
<td>\xtShowTerminalNodes</td>
<td>0.8</td>
</tr>
<tr>
<td>\HideTerminalNodes</td>
<td>\xtHideTerminalNodes</td>
<td>0.8</td>
</tr>
<tr>
<td>\levdist</td>
<td>\xtlevdist</td>
<td>1.0</td>
</tr>
<tr>
<td>\sibdist</td>
<td>\xtsibdist</td>
<td>1.0</td>
</tr>
<tr>
<td>\setistactionlabelshift</td>
<td>\xtALPush</td>
<td>1.0</td>
</tr>
<tr>
<td>\setistactionlabelposition</td>
<td>\xtALShift</td>
<td>1.0</td>
</tr>
<tr>
<td>\istroot*</td>
<td>\istrooto</td>
<td>1.0</td>
</tr>
<tr>
<td>\istcntm*</td>
<td>\istcntmarc</td>
<td>1.0</td>
</tr>
<tr>
<td>\xtInfoset*</td>
<td>\xtInfoseto</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Here, the prefix ‘xt’ stands for ‘extensive tree,’ so \xtdistance can be read as ‘extensive tree distance.’

Remark: A known problem with the tikz-qtree package:

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

The best way to resolve this problem is that you DO NOT LOAD tikz-qtree when you draw game trees with TikZ. If, for some reason, you need to load tikz-qtree when you draw a game tree by using the istgame package, a temporary solution to resolve the conflict is to redefine the istgame environment as follows:

\begin{Verbatim}
\RenewDocumentEnvironment{istgame}{O{}} % accepts tikzpicture options
  \begin{tikzpicture}\%
    edge from parent path={([tikzparentnode\istparentanchor] -- \tikzchildnode\istchildanchor)}, % tikz-qtree conflict resolved
    font=\istgamefontsize,>=stealth,#1
  \end{tikzpicture}
\end{Verbatim}

1The istgame package of the version which is older than ver. 1.0 had been distributed in the KTUG (Korean TeX Users Group).
1 Getting started

The package istgame provides macros built on TikZ to draw game trees. The core macros provided with this package are \istroot, \istb, and \endist. \istroot pins down the root of a tree or a subtree, \istb represents a branch, and \endist indicates the end of drawing a simple tree. Without \endist, the tree is not actually drawn, with no error messages produced. A tree drawn by the sequence of \istroot-\istb-\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:

\usepackage{istgame}

The package loads the following TikZ libraries:
calc, arrows, shapes, positioning, patterns, trees, fit, backgrounds, quotes

1.1 Getting-started example: a simple tree

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

```latex
\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}
```

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

If the second parenthesis argument of \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 \istroot-\istb-\endist structure.

```latex
\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)
\end{istgame}
```

In the previous 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 given names of the (sub)roots and the names of endpoints are given counterclockwise (from the left to the 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 from the decision node with action labels and payoffs, and \texttt{\textbackslash endist} actually draws the tree structures.

\begin{verbatim}
\begin{istgame}
\xdistance{15mm}{30mm}
\istroot(0)(0,0){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}
\end{verbatim}

1.3.1 How to put a decision node and its owner

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

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

In fact, \texttt{\textbackslash istroot} and its variations have much more functions than these. Later, you can look into section 4 for more details.

1.3.2 How to put action labels and payoffs

\texttt{\textbackslash istb\{<action label>\}\{<action pos>\}\{<payoffs>\}\{<payoff pos>\}}

With the macro \texttt{\textbackslash istb}, you can put a action label and payoffs as optional arguments. The positions of an action label and payoffs are specified as options right after each of the two.

In fact, \texttt{\textbackslash istb} and its variations have much more functions than these. See section 5 for more details. If this is your first reading this manual, however, you don’t need to bother about all the details at the moment.

1.3.3 How to put information sets

\texttt{\textbackslash xtInfoset\{<from coor>\}\{<end coor>\}\{<owner>\}}

The macro \texttt{\textbackslash xtInfoset} connects two nodes with a dotted line. The two node coordinates are mandatory. For more details about \texttt{\textbackslash xtInfoset}, see section 9.

In the following example, the macro \texttt{\textbackslash xtInfoset} is used to show a information set. \texttt{\textbackslash al} is an abbreviation of \texttt{\{above left\}}, and similarly for \texttt{\{ar\}}, \texttt{\{bl\}}, and \texttt{\{br\}}.
2 Important distances: \texttt{\textbackslash xtdistance}

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

\texttt{\textbackslash xtdistance} sets or resets the level and sibling distances. You can use \texttt{\textbackslash xtdistance} at any time you want to change the length and the direction of branches. Since we are dealing with simple ‘parent-child’ tree structures, \texttt{\textbackslash level depth} is 1 by default. However, the level depth number other than 1 is not expected to be used.

For example, \texttt{\textbackslash xtdistance{20mm}{30mm}} internally assigns 20mm to \texttt{\xtlevdist} and 30mm to \texttt{\xtsibdist}, which renews the default distances.

In fact, the core macros are much more powerful. \texttt{\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. \texttt{\istem} specifies the growing direction of an individual branch, branch line styles, branch color, the label of action, and the payoffs and their positions. Below we will see in more detail how the core macros and others work.
3 The istgame environment and node styles

3.1 The istgame environment

This package provides the istgame environment, which is almost the same as \texttt{tikzpicture} environment, so this accepts all the options of the \texttt{tikzpicture} environment.

\begin{Verbatim}
\def\istgamefontsize{\normalsize}
\NewDocumentCommand\setistgamefontsize{m}
  \{\renewcommand*{\istgamefontsize}{#1}\}
\NewDocumentEnvironment{istgame}[0] % accepts tikzpicture options
  \begin{tikzpicture}[font=\istgamefontsize,>=stealth,#1]
  \end{tikzpicture}
\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 istgame is basically the same as tikzpicture, you can also locally change the font size by using the \texttt{font} option key, like \texttt{\begin{istgame}[font=\scriptsize]...\end{istgame}}.

3.2 Node styles

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

- plain node: draws nothing (defaults: \texttt{inner sep=1pt, outer sep=0pt})
- null node: . (very small node, not expected to be used)
- solid node: . (default node style)
- hollow node: ◯
- rectangle node: \[
- ellipse node: ○

For some special cases, you may want to change some node styles, including the minimum size. This can be done by \texttt{\setist<..>NodeStyle}, all of whose arguments are optional.

\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}

\begin{Verbatim}
% Examples:
\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[red]{10pt}[.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
\tikzstyle{decision node}=[solid node] % decision nodes
\tikzstyle{terminal node}=[solid node] % terminal nodes
\tikzstyle{initial node}=[hollow node]
\tikzstyle{chance node}=[hollow node]

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
\tikzstyle{box node}=[rectangle node]
\tikzstyle{square node}=[rectangle node]
\tikzstyle{oval node}=[ellipse node]

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

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 (sub)root, and does other functions. In game theoretic terminology, \istroot designates a decision node and its owner (or a player).

% \istroot
% syntax:
\istroot[<grow keyval>,<opt>](<coor1>)(<coor2>)[<node style>,<opt>]
% <owner label angle>{<owner>}+<lev-distance>..<sib-distance>+
% defaults:
[<south>] (0,0)[decision node]above}{15mm..15mm+<angle>}{<owner>}{<sibling distance>+
% arguments: (coor1) in 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
+level dist..sibling dist+ % <default: 15mm,15mm>

The only mandatory argument of \istroot is <<coor1>>, which gives the name of the root or subroot. All the other arguments are optional. The name of the (sub)root, <<coor1>>, can be referred as a normal coordinate. <<coor2>> specifies the location where the (sub)root is placed. If <<coor2>> is omitted, it is regarded as (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, for example, an oval node or a box node.

The owner of a decision node (or a player) is expressed with curly braces. The position of the owner of a decision node is specified with angle brackets, like <90>, <above>, or <north>. To specify the position of the owner you can use <degrees>, or the compass directions such as
<north>, <south>, <east>, <west>, and their valid combinations. You can also use the positional words such as <above>, <below>, <left>, <right>, and their valid combinations.

Here is a simple example of drawing a tree structure.

```
\begin{istgame}
  \istroot(0)
  \istb \istb \istb \endist
  \istroot(a)(0-1)
  \istb \istb \endist
\end{istgame}
```

In the previous example, the game tree has the root named (0), located at (0,0) by default, which has three branches. Since TikZ arranges branches of a tree counterclockwise, by default, the endpoints of the three branches are automatically named (0-1), (0-2), and (0-3) from left to right or counterclockwise.

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.

```
\begin{istgame}[font=\scriptsize]
  \istroot(0)
  \istb \istb \istb \endist
  \istroot(a)(0-1)
  \istb \istb \endist
  \xtOwner(0){(0)}
  \xtOwner(0-1){(0-1)}[r]
  \xtOwner(0-2){(0-2)}[b]
  \xtOwner(0-3){(0-3)}[b]
  \xtOwner(a){(a)}[1]
  \xtOwner(a-1){(a-1)}[b]
  \xtOwner(a-2){(a-2)}[b]
  \draw[ultra thick,blue!20](-260:1.5) arc (-260:80:1.5cm)
  \node[above,blue!30] {counterclockwise};
\end{istgame}
```

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:** For some cases, you may want to use options other than the grow direction, in the bracket before \texttt{(coor1)}. In this case, the first thing in the bracket must be \texttt{<grow keyval>}, like \texttt{[south,blue,dashed]} (see section 6.1.2 on page 14).

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

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 \texttt{[above]} nor \texttt{[below]}.
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, `[below]` and `[above]` are good, but not `[down]` nor `[up]`. So DO NOT USE `[down]` and `[up]` to specify the tree growing direction.

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

The last two options of `\istroot` specify the level distance and the sibling distance. This change of distances is valid only for the corresponding simple tree, while distance change by `\xtdistance` is valid within the current `\istgame` environment unless it is 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` on page 7. The two examples have exactly the same codes as each other except for one thing: either `\istroot` or `\istroot'`.

\begin{verbatim}
% Example: \istroot' (explained with labels)
\begin{istgame}[font=\scriptsize]
\istroot'(0)
\\\istb \\\istb \\\istb \\\endist
\istroot'(a)(0-1)
\\\istb \\\istb \\\istb \\\endist
\% labels: (ignore the following lines at the moment)
\xtOwner(0){(0)}
\xtOwner(0-1){(0-1)}[r]
\xtOwner(0-2){(0-2)}[b]
\xtOwner(0-3){(0-3)}[b]
\xtOwner(a){(a)}[1]
\xtOwner(a-1){(a-1)}[b]
\xtOwner(a-2){(a-2)}[b]
\\draw [-,ultra thick,blue!20](-260:1.5)
\\arc (-260:80:1.5cm)
\\node [above,blue!30] {\{\text{\texttt{clockwise}}\};
\end{istgame}
\end{verbatim}

We will look into this issue (of going counterclockwise or clockwise) in more detail in section 8 where we examine the tree growing direction.

If you draw a game tree growing south, you don’t need to worry about the swap version `\istroot'` and just use `\istroot`.

4.2 `\istrootoo`: oval version

4.2.1 `\istrootoo` – counterclockwise

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

\begin{verbatim}
\% \istroot
\% syntax:
\istroot{[<grow keyval>]>({<coor1>},{<coor2>})[<node style>])<player>]+levd..sibd+
\% default: only (coor1) is mandatory, all others optional
[south](0,0)[oval node]{+15mm..15mm+}
\end{verbatim}

The following example is the same as above with `\istroot` on page 7, but now with the oval version.

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

The following example is also the same as above with `\istroot` on page 8, but now with the oval version.

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

Notice that the angle `<180>`, `<left>`, `<right>`, or `<135>` specifying the position of the owner’s name is redundant to `\istroot`.

\textbf{Remark:} Note that, in Ti\TeX, a tree branch comes from the center of the parent node and goes to the center of the child node.

Note also that `\istroot` arranges its branches \textit{counterclockwise}.  

\begin{verbatim}
\% Example 3: \istroot (counterclockwise)
\begin{istgame}
\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]\istb{}{5}[center]\endist
\istroot(2)(0-2){2}\istb{}{6}[center]\istb{}{7}[center]\istb{}{8}[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 [above,blue!30] {counterclockwise};
\end{istgame}
\end{verbatim}
4.2.2 \texttt{\istroot'} – clockwise (swap version)

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

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

The swap version is useful when a tree grows north or east. The following example shows the tree rotating to the north by using \texttt{\setistgrowdirection}. Note that \texttt{\istroot'} arranges its branches clockwise.

\begin{verbatim}
\% Example 3: \istroot' (clockwise)
\begin{istgame}
\setistgrowdirection{north}
\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]
\istb{5}[center] \endist
\istroot'(2)(0-2){2}
\istb{6}[center] \istb{7}[center]
\istb{8}[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}
\end{verbatim}

5 Core macro: \texttt{\istb}

5.1 \texttt{\istb}: basics

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

The macro \texttt{\istb} also puts an action label and payoffs along with a branch, and does other functions. Note that the action labels and payoffs are defined to be typeset in the math mode, so you can use \texttt{\text{}\ldots{}} when you want to typeset them in the text mode.
In the following example, each `\istb` draws a branch. With the option `<missing>`, `\istb` prints an invisible branch. Since the third child is missing the last child was named (0-4).

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

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

**Remark:** What is `\istgrowdirection` and what is it used for?

`\istgrowdirection` has the value of `[<grow keyval>]` typed in `\istroot` (default: `south`). The value of `\istgrowdirection` specifies the direction of putting ‘payoffs’ with `south` by default. We will see this in more detail below.

For various line styles of a branch, you can use any options in Ti\kZ of arrows, linewidth, color, and so on.

```
\begin{istgame}
\istroot(0)
\istb[dashed, thick]{A}{(0,1)}
\istb[->]{(1,-1)}
\istb[blue, very thick]{B}[above]{(1,2)}
\end{istgame}
```

By default, an action label is put on the mid point of the corresponding branch.

To specify the position of an action label, you can use the positional words or their abbreviations below:

```latex
[a] for [above], [b] for [below], [l] for [left], [r] for [right],
[al] for [above left], [ar] for [above right],
[bil] for [below left], and [ibr] for [below right].
```

**Remark:** Note that these abbreviations must be used with no other options, otherwise you will get a compile error.

In the following example, notice that the tree grows south east and the payoffs are placed south east.
Example: tree toward south east (or -45 degree)
\begin{ngame}
\istroot[-45](0)
\istb[dashed,thick]{A}{{(0,1)}}
\istb[->]{{(1,-1)}}
\istb[blue,very thick]{B}{above}{{(1,2)}}
\endist
\end{ngame}

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

\begin{ngame}
\istroot[south west](0)
\istb[dashed,thick]{A}{{(0,1)}}
\istb[->]{{(1,-1)}}
\istb[blue,very thick]{B}{right}{{(1,2)}[below]}
\endist
\end{ngame}

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

Each endpoint is printed by each execution of \istb*. You can print solid nodes (by default) at ‘all’ the endpoints of ‘simple trees’ by \xtshowendpoints. You can also change the shape of nodes of endpoints of simple trees by specifying it as an optional argument, like \xtshowendpoints[oval node]. \xthideendpoints turns off the effect of \xtshowendpoints.

\begin{ngame}
\istroot[ east](0)
\istb[dashed,thick]{A}
\istb[->]{{(1,-1)}}
\istb[blue,very thick]{B}{above}
\endist
\end{ngame}

\xtshowendpoints and \xthideendpoints

Here is an example of using \xtshowendpoints and \xthideendpoints.
5.3 \istb. (period version)

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

\xtShowTerminalNodes and \xtHideTerminalNodes

The period version \istb. used with \xtShowTerminalNodes prints a solid node (by default). You can change the style of the terminal nodes, like \xtShowTerminalNodes[oval node]. This effect can be turned off by \xtHideTerminalNodes.

6 Players, action labels, and payoffs

6.1 How to put players

6.1.1 Players: basics

As discussed in 4.1.1, the macro \istroot specifies the (sub)root and puts its owner (or a player). The direction to which a player label is put is set by the angle \ang option of \istroot, like \ang{above}
(by default), <0>, or <45>. To specify the direction you can use degrees, the compass directions, or the positional words.

\begin{verbatim}
% Example: node owner
\begin{tikzpicture}
  \node (owner) [minimum size=1cm] {
    \tikzset{\ nodesize}
  };
  \draw (0,0) node [above] {Child};
  \node (parent) at (0,2) [minimum size=1cm] {
    \tikzset{\ nodesize}
  };
  \draw (0,2) node [above] {Parent};
\end{tikzpicture}
\end{verbatim}

Remark: The auxiliary macro \xtOwner is provided for an extra way of putting owners of decision nodes (see section 11).

Note that \tikzroot produces a bubble type node with the owner in it, so the directional option \ang is redundant with \tikzroot (see section 4.2).

\begin{verbatim}
% Example: node owner (with \tikzroot)
\begin{tikzpicture}
  \node (owner) [minimum size=1cm] {
    \tikzset{\ nodesize}
  };
  \draw (0,0) node [above] {Child};
  \node (parent) at (0,2) [minimum size=1cm] {
    \tikzset{\ nodesize}
  };
  \draw (0,2) node [above] {Parent};
\end{tikzpicture}
\end{verbatim}

6.1.2 Coloring players or a whole simple tree: \tikzroot

You can change the color of a player’s name by giving color with the angle option of \tikzroot, like <[red]> or <[blue]45>. (This is the TikZ way for giving options for label.)

\begin{verbatim}
% Example: coloring players
\begin{tikzpicture}
  \tikzset{\nodesize}
  \node (owner) [minimum size=1cm] {
    \node [red] {Child};
  };
  \draw (0,0) node [above] {Child};
  \node (parent) at (0,2) [minimum size=1cm] {
    \node [blue] {Parent};
  };
  \draw (0,2) node [above] {Parent};
\end{tikzpicture}
\end{verbatim}

The following example shows how you color a whole simple tree: red for the Child’s simple tree and blue for the Parent’s simple tree.

Remark: Note that the first thing in the \option of \tikzroot must be ‘the direction of tree growing,’ so it should be set as [south, red] or [-90, blue].

\begin{verbatim}
% Example: coloring a simple tree
\begin{tikzpicture}
  \tikzset{\nodesize}
  \node (owner) [minimum size=1cm] {
    \node [red] {Child};
  };
  \node (parent) at (0,2) [minimum size=1cm] {
    \node [blue] {Parent};
  };
\end{tikzpicture}
\end{verbatim}
6.1.3 Decorating players or a whole simple tree: \texttt{istroot}

With \texttt{istroot} the color of a player can be changed by [option], such as \texttt{[red]} or \texttt{[blue]}. (This is the Ti\kZ way for giving options for node.)

\begin{verbatim}
% Example: coloring players (with \texttt{istroot})
\begin{istgame}
\istroot(0)[red]{Child}
 \istb \istb \endist
\istroot((0-2)[blue]{Parent}
 \istb \istb \endist
\end{istgame}
\end{verbatim}

The next example shows how you can color a whole simple tree \texttt{red} or \texttt{blue}.

\begin{verbatim}
% Example: coloring a simple tree (with \texttt{istroot})
\begin{istgame}
\istroot[south,red](0)[draw=green,blue]{Child}
 \istb \istb \endist
\istroot[-90,blue](1)(0-2)[draw=blue,blue]{Parent}
 \istb \istb \endist
\end{istgame}
\end{verbatim}

What if you want to paint some color into the background of each oval node or box node? You can also do this simply by using the Ti\kZ way for giving options for nodes.

\begin{verbatim}
% Example: decorating oval/box nodes
\begin{istgame}
\istroot(0)[top color=white, bottom color=blue!80!black,yellow]{Child}
 \istb \istb \endist
\istroot((0-2)[box node,fill=red!20,blue]{Parent}
 \istb \istb \endist
\end{istgame}
\end{verbatim}

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 are printed in the math mode.

\begin{verbatim}
% Example: actions labels
\begin{istgame}
\istroot(0)
 \istb{Good}{left} \istb{Bad}{right} \endist
\istroot((0-2)
 \istb{\text{\fbox{$\alpha$}}}{above left} \istb{\text{\fbox{$\beta$}}}{above right} \endist
\end{istgame}
\end{verbatim}

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

Notice that with the abbreviations the position of the action labels are (internally) adjusted to get better result (for more details, see section \ref{section:7.2.2}). In the following example, the abbreviations are used.
\begin{example}
\begin{verbatim}
\begin{istgame}
  \textbf{istroot}(0)
  \istb{Good}[1] \istb{Bad} \endist
  \istroot(1)(0-2)
  \istb{\text{\fbox{$\alpha$}}}[al]
  \istb{\text{\fbox{$\beta$}}}[ar]
  \endist
\end{istgame}
\end{verbatim}
\end{example}

Remark: The auxiliary macro \xtActionLabel is provided for an extra way of putting an action label (see section \ref{sec:xtActionLabel}).

6.2.2 Changing the color of action labels and the style of branches

You can change the color of action labels with TikZ options. In the following example, the bracket options before an action label are for a branch and those after are only for the action label.

\begin{example}
\begin{verbatim}
\begin{istgame}
  \textbf{istroot}(0)
  \istb[dashed,thick]{A}[al]
  \istb[blue,very thick]{B}[r]
  \istb[blue,very thick]{C}[above right,red]
  \endist
\end{istgame}
\end{verbatim}
\end{example}

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

6.3 How to put payoffs

6.3.1 Payoffs: basics

The macro \istsb can also print a branch and the corresponding payoffs near at its endpoint. Note also that payoffs are printed in the math mode. The payoffs are put in the direction set by \setistgrowdirection (south, by default), unless it is changed by \growkeyval of \istroot.

\begin{example}
\begin{verbatim}
\begin{istgame}
  \textbf{istroot}(0)
  \istsb{Good}[1]{0,2}
  \istsb{Bad}
  \endist
  \istsroot(1)(0-2)
  \istsb{\alpha}[al]{(1,1)}
  \istsb{\beta}[ar]{-1,-1}
  \endist
\end{istgame}
\end{verbatim}
\end{example}

6.3.2 Payoffs and \istgrowdirection

The direction of where payoffs are put from a terminal node follows \istgrowdirection typed in as the first optional argument of \istroot. The default direction is south and can be changed by \setistgrowdirection. For example, \setistgrowdirection{north} changes the default direction to north.
To specify the tree growing direction or the position of the payoffs to put, you can use degrees, the compass words, or the positional words and their abbreviations.

% Example: grow=south (default)
\begin{istgame}
\xtShowEndPoints[box node]
\istroot(0)
\\istb[dashed,thick]{$A$}\{\binom{2}{3}}
\\istb*{\{(-2,3)}
\\istb.[blue,very thick]{$B$}[ar]{2,3}
\endist
\end{istgame}

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

% Example: grow=right (or east or 0 degree)
\begin{istgame}
\xtShowTerminalNodes[oval node]
\istroot[right](0)
\\istb[dashed,thick]{$A$}\{\binom{2}{3}}
\\istb*{\{(-2,3)}
\\istb.[blue,very thick]{$B$}[al]{2,3}
\endist
\\xtHideTerminalNodes
\end{istgame}

grow=right=\texttt{\istgrowdirection}, so the payoffs are put on the right.

% Example: grow=north
\begin{istgame}
\setistgrowdirection{north}
\istroot(0)
\\istb[dashed,thick]{$A$}\{\binom{2}{3}}
\\istb*{\{(-2,3)}
\\istb.[blue,very thick]{$B$}[bl]{2,3}
\endist
\end{istgame}

grow=north=\texttt{\istgrowdirection}, so the payoffs are put above the terminal nodes.

\begin{istgame}
\istroot[south west](0)
\\istb[dashed,thick]{$A$}\{\binom{2}{3}}
\\istb*{\{(-2,3)}
\\istb[blue,very thick]{$B$}[br]{2,3}{b}
\endist
\end{istgame}

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

You can adjust the direction of putting any payoff by specifying an option right after the payoff, like \texttt{\istb[blue,very thick]{$B$}[right]{2,3}[below]}.

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 7.2.2).
Notice also that, instead of the positional words, you can use the compass directions or degrees, like \texttt{\istb[blue,very thick]{B}[right]{2,3}[-90]}.

6.3.3 Coloring payoffs

You can change the color of the payoffs by giving options with the positional words. For example you can do like \texttt{\istb...{(1,1)}[blue]below}. This is the TikZ way of giving options for \texttt{label}. Notice that, in this case, you should not use the abbreviation of the positional words.

\begin{verbatim}
% Example: payoffs
\begin{istgame}
\istroot(0)
\istb(Good){0,2} \istb(Bad) \endist
\istroot(1){0-2}
\istb(\alpha){al}{(1,1)}[red]below
\istb(\beta){ar}{-1,-1}[blue]below
\endist
\end{istgame}
\end{verbatim}

7 Fine-tuning positions of players, action labels, and payoffs (experimental)

7.1 Fine-tuning positions: owners

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

Examples are \texttt{<[xshift=10pt]above>{Child}} and \texttt{<[label distance=-5pt]45>{Parent}}, as shown in the following.

\begin{verbatim}
% Example: node owner
\begin{istgame}[scale=1.2]
\xtShowEndPoints
\xtdistance{12mm}{16mm}
\istroot(0)[initial node]
\istb<grow=0>{\textbf{fbox}{$a$}}[a]
\istb<grow=90>{\textbf{fbox}{$l$}}[l]
\istb<grow=180>{\textbf{fbox}{$b$}}[b]
\istb<grow=-90>{\textbf{fbox}{$r$}}[r]
\endist
\end{istgame}
\end{verbatim}

7.2 Fine-tuning positions: action labels

7.2.1 Abbreviations: [l], [r], [a], and [b]

As discussed in \textsection{5.1}, the macro \texttt{\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 their abbreviations as mentioned above. (Internally, the abbreviations for payoffs and those for action labels work slightly differently in terms of \texttt{xshift} and \texttt{yshift}.)

\begin{verbatim}
% Example: action labels (default position)
\begin{istgame}[scale=1.2]
\xtShowEndPoints
\xtdistance{12mm}{16mm}
\istroot(0)[initial node]
\istb<grow=0>{\textbf{fbox}{$a$}}[a]
\istb<grow=90>{\textbf{fbox}{$l$}}[l]
\istb<grow=180>{\textbf{fbox}{$b$}}[b]
\istb<grow=-90>{\textbf{fbox}{$r$}}[r]
\endist
\end{istgame}
\end{verbatim}
When you use these abbreviations you can manipulate the horizontal and/or the vertical shifts toward branches by using \xtALPush. (This is experimental!)

\begin{verbatim}
\% syntax:  
\xtALPush{<xshift dim> for l and r}{<yshift dim> for a and b}  
\% default:  
{0pt}{0pt}
\end{verbatim}

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.

\begin{verbatim}
\% Example: \xtALPush
\begin{istgame}[scale=1.2]
\xtShowEndPoints
\xtALPush{-3pt}{5pt} \% look here
\xtdistance{12mm}{16mm}
\istroot(0)[initial node]
\text{a}
\text{b}
\end{istgame}
\end{verbatim}

7.2.2 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{verbatim}
\% Example: action labels with abbreviations
\begin{istgame}[scale=1.1]
\xtShowTerminalNodes[oval node, line width=.3pt]
\def\xbox#1{\fbox{$#1$}} \def\ybox#1{#1}
\xtdistance{12mm}{16mm}
\istroot(0)[initial node]
\istb[0\times90]{\xbox{al}}
\istb[180\times90]{\xbox{bl}}
\end{istgame}
\end{verbatim}
In Figure 1, \( al \)'s are put in the same position from dotted branches, and so are the other labels from their corresponding branches, like \( bl \)'s from the blue dashed branches.

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

When the dimensions get bigger than the defaults (1pt and 2pt) the labels get closer to the mid points of the corresponding branches, and when the numbers get smaller the labels get farther from their branches.
7.3 Fine-tuning positions: payoffs

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

\begin{verbatim}
% Example: payoffs
\begin{istgame}
  \istroot(0)
  \istb{Good}[1]{0,2} \istb{Bad} \endist
  \istb{\alpha}[al]{{(1,1)}[xshift=-10pt]below}
  \istb{\beta}[ar]{-1,-1}[yshift=-10pt]below
  \endist
\end{istgame}
\end{verbatim}

8 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 below to above.

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 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’}.

\begin{verbatim}
% default: growing south counterclockwise
% \def\xtgrow{grow}
% \def\istdefault@grow[south] % tree growing direction
% \% \setistgrowdirection()
% NewDocumentCommand\setistgrowdirection{t'm}
% {\IfBooleanTF {#1}
%  {\renewcommand\xtgrow{grow'}
%   \renewcommand\istdefault@grow[#2]
%  }{\renewcommand\xtgrow{grow}
%   \renewcommand\istdefault@grow[/#2]
%  }
%  }
%  }
\end{verbatim}

8.1 \texttt{\setistgrowdirection} – counterclockwise

Our first example is a tree drawn using the default values: growing south with branches going counterclockwise with respect to their parent nodes (from left to right).
By default or with \setistgrowdirection{south}, the numbers written in the child nodes increase counterclockwise with respect to their parent nodes.

8.2 \setistgrowdirection' – clockwise

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

8.3 Examples of rotating trees with \setistgrowdirection(')

This macro allows you 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 \setistgrowdirection'.

Tips:

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

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

8.3.1 A tree growing east – counterclockwise
% Example 3: \setistgrowdirection{east}
\begin{istgame}
\setistgrowdirection{east}
% same code 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(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 upward (or counterclockwise).

### 8.3.2 A tree growing east – clockwise

% Example 4: \setistgrowdirection{east}
\begin{istgame}
\setistgrowdirection{east}
% same code 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}

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

### 8.3.3 A tree growing north – counterclockwise

% Example 5: \setistgrowdirection{north}
\begin{istgame}
\setistgrowdirection{north}
% same code 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).
8.3.4 A tree growing north - clockwise

Example 6:

\begin{istgame}
\setistgrowdirection\{north\}
\setistOvalNodeStyle\{.6cm\}
\xtShowEndPoints\{oval node\}
\xtdistance\{12.5mm\}\{11.5mm\}
\istroot\{0\}\{0+{12.5mm}..{3.45cm}\}+
\istb \istb \endist
\xtdistance\{12.5mm\}\{11.5mm\}
\istroot\{1\}\{0-1\}\{1\}
\istb\{3\}\{center\} \istb\{4\}\{center\}
\istb\{5\}\{center\} \endist
\istroot\{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.

8.3.5 \setistgrowkey for one simple tree

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

\begin{Verbatim}
\% \setistgrowkey: definition
\NewDocumentCommand\setistgrowkey\{m\}
{ \renewcommand\xtgrow{#1}\{grow\}}
\% #1 is either grow or grow'
\end{Verbatim}

In the following example, by using \setistgrowdirection' the branches are arranged clockwise. So you will need to use \setistgrow\{grow\} to locally get back to \{counterclockwise\}.

\begin{Verbatim}
\% Example: using \setistgrowdirection' (swap version)
\begin{istgame}
\setistgrowdirection\{east\}
\istroot\{0\}
\istb\{a\}\{al\} \istb\{b\}\{bl\} \endist
\istroot\{north\}\{1\}\{0-1\}
\istb\{c\}\{bl\} \istb\{d\}\{br\} \endist
\{\setistgrowkey\{grow\}\}
\istroot\{south\}\{2\}\{0-2\}
\istb\{e\}\{al\} \istb\{f\}\{ar\} \endist
\end{istgame}
\end{Verbatim}

You can do the same thing by using \setistgrowdirection and \istroot'. See the following example, in which you do not need to use \setistgrowkey.
9 Information sets

9.1 Information sets: \texttt{\textbackslash xtInfoset()} 

The macro \texttt{\textbackslash xtInfoset} draws an information set, connecting two decision nodes.

\begin{verbatim}
% Example: using \textbackslash istroot' (swap version)
\begin{istgame}
\setistgrowdirection(east)
\istroot'(0)
\istb{a}[al] \istb{b}[bl] \endist
\istroot'[north](1)(0-1)
\istb{c}[bl] \istb{d}[br] \endist
\istroot[south](2)(0-2)
\istb{e}[al] \istb{f}[ar] \endist
\istroot'(3)(2-2)
\istb{g}[al] \istb{h}[bl] \endist
\end{istgame}
\end{verbatim}

The swap version \texttt{\textbackslash xtInfoset'} is provided, which is not very interested.

\begin{verbatim}
% \texttt{\textbackslash xtInfoset}
% syntax: from left to right
\xtInfoset*[info line type]*(<from>)(<to>){<owner>}[<pos>,<node opt>]
% defaults: [semithick,dotted]()(){above]
\end{verbatim}

\begin{verbatim}
% Example: \texttt{\textbackslash xtInfoset}
\begin{istgame}
\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
\xtInfoset(1)(2){2}
\end{istgame}
\end{istgame}
\end{verbatim}

\texttt{\textbackslash xtInfoset'} connects two nodes from the coordinate (\texttt{<to>}) to the coordinate (\texttt{<from>}), while \texttt{\textbackslash xtInfoset} from the coordinate (\texttt{<from>}) to the coordinate (\texttt{<to>}). So, basically, the two macros give the same result, without arrow tips.
9.2 Information sets: \xtInfosetO(\') (experimental)

The oval version \xtInfosetO prints a bubble type information set. (experimental!)

\xtInfosetO prints a dotted rectangle with rounded corners, connecting two nodes. The size of the information set can be changed by the \(\langle\text{sep}\rangle\) option, whose default value is 4pt.

You can change the color or the size of the information set by using options, as shown in the following example:
The swap version \texttt{\textbackslash xtInfosetO'} is also provided, which you should be aware of.

When do we need the swap version \texttt{\textbackslash xtInfosetO'}? Let’s look at the following example, where you would not be satisfied with how the information sets look.

The rule of thumb for using \texttt{\textbackslash xtInfosetO'} is this: whenever the bubbled information set turns out to be short, use the swap version. You can use \texttt{\textbackslash xtInfosetO} and \texttt{\textbackslash xtInfosetO'} interchangeably, to get correct results. More precisely, if you follow the tips about the combinations of \texttt{\setistgrowdirection(')} and the compass directions, as suggested on page 22, you will get correct results. However, if you use the combinations other than the suggestions, you will probably get the short information set. Then, use \texttt{\textbackslash xtInfosetO'}.

\section{Continuum of branches}

The package istgame provides two types of macros to represent a continuum of branches: \texttt{\textbackslash istcntm} and \texttt{\textbackslash istcntmarc}. Each of the two macros, basically, works like \texttt{\textbackslash istroot}, but just draws a background to express a continuum of branches.

\subsection{\texttt{\textbackslash istcntm} (standard version)}

\subsubsection{Basic}

The first type macro \texttt{\textbackslash istcntm} is the standard version. This, in fact, draws a background with the optional color filled representing a continuum of branches (the default color is black!25). The
shape of the background is a isosceles triangle, with the height of \texttt{cntmlevdist} (8mm by default) and the base of \texttt{cntmsibdist} (initially 3*8mm or 24mm by default).

\begin{verbatim}
\% \texttt{\textbackslash istcntm} \\
\% syntax: \\
\texttt{\textbackslash istcntm}[\texttt{grow keyval}](<coor1>)(<coor2>)[<fill color>]+<levdist>..<sibdist>+ \\
\% defaults: \\
[\texttt{south}](0,0)[\texttt{black!25}]+8mm..3*8mm+
\end{verbatim}

Using \texttt{\textbackslash istroot} with \texttt{\textbackslash istcntm} draws a representative branch of a continuum of branches. Note, in the following example, that the second branch of \texttt{\textbackslash istroot} is missing to express the representative branch among a continuum of branches. Again, \texttt{cntmlevdist} is 8mm by default.

\begin{verbatim}
\begin{center}
\begin{tikzpicture}
\node[draw=none, inner sep=0] at (0,0) {I};
\node[draw=none, inner sep=0] at (1,0) {II};
\node[draw=none, inner sep=0] at (1.5em,0) {x};
\node[draw=none, inner sep=0] at (1.5em,1.5em) {\texttt{Y}};
\node[draw=none, inner sep=0] at (0,1.5em) {\texttt{N}};
\draw[<-,very thick] (0,0) -- (1,0) node[midway,above] {x};
\draw[<-,very thick] (0,0) -- (1.5em,0) node[midway,above] {\texttt{X}};
\draw[<-,very thick] (1,0) -- (1.5em,1.5em) node[midway,left] {\texttt{Y}};
\draw[<-,very thick] (0,0) -- (0,1.5em) node[midway,right] {\texttt{N}};
\end{tikzpicture}
\end{center}
\end{verbatim}

10.1.2 Changing the size and the color of a continuum of branches

You can change the height (or the level distance) and the base (or the sibling distance) of the background isosceles triangle by specifying them, for example, like +10mm..20mm+.

\begin{verbatim}
\begin{verbatim}
\% Example: \texttt{\textbackslash istcntm} \texttt{(changing the size)} \\
\begin{center}
\begin{tikzpicture}
\node[draw=none, inner sep=0] at (0,0) {I};
\node[draw=none, inner sep=0] at (1,0) {II};
\node[draw=none, inner sep=0] at (1.5em,0) {x};
\node[draw=none, inner sep=0] at (1.5em,1.5em) {\texttt{Y}};
\node[draw=none, inner sep=0] at (0,1.5em) {\texttt{N}};
\draw[<-,very thick] (0,0) -- (1,0) node[midway,above] {x};
\draw[<-,very thick] (0,0) -- (1.5em,0) node[midway,above] {\texttt{X}};
\draw[<-,very thick] (1,0) -- (1.5em,1.5em) node[midway,left] {\texttt{Y}};
\draw[<-,very thick] (0,0) -- (0,1.5em) node[midway,right] {\texttt{N}};
\end{tikzpicture}
\end{center}
\end{verbatim}
\end{verbatim}

Once the distances are type in as options of \texttt{\textbackslash istcntm}, they internally change the default values, which apply only to the corresponding triangle. For example, +10mm..20mm+ will result in \texttt{cntmlevdist}=10mm and \texttt{cntmsibdist}=20mm. So you can get exactly the same result as previous one by doing the following:

\begin{verbatim}
\% Example: \texttt{\textbackslash istcntm} \texttt{(changing the size)} \\
\begin{center}
\begin{tikzpicture}
\node[draw=none, inner sep=0] at (0,0) {I};
\node[draw=none, inner sep=0] at (1,0) {II};
\node[draw=none, inner sep=0] at (1.5em,0) {x};
\node[draw=none, inner sep=0] at (1.5em,1.5em) {\texttt{Y}};
\node[draw=none, inner sep=0] at (0,1.5em) {\texttt{N}};
\draw[<-,very thick] (0,0) -- (1,0) node[midway,above] {x};
\draw[<-,very thick] (0,0) -- (1.5em,0) node[midway,above] {\texttt{X}};
\draw[<-,very thick] (1,0) -- (1.5em,1.5em) node[midway,left] {\texttt{Y}};
\draw[<-,very thick] (0,0) -- (0,1.5em) node[midway,right] {\texttt{N}};
\end{tikzpicture}
\end{center}
\end{verbatim}

You can change the color of the background triangle by using an option as you can see in the following example:

\begin{verbatim}
\% Example: \texttt{\textbackslash istcntm} \texttt{(changing the size)} \\
\begin{center}
\begin{tikzpicture}
\node[draw=none, inner sep=0] at (0,0) {I};
\node[draw=none, inner sep=0] at (1,0) {II};
\node[draw=none, inner sep=0] at (1.5em,0) {x};
\node[draw=none, inner sep=0] at (1.5em,1.5em) {\texttt{Y}};
\node[draw=none, inner sep=0] at (0,1.5em) {\texttt{N}};
\draw[<-,very thick] (0,0) -- (1,0) node[midway,above] {x};
\draw[<-,very thick] (0,0) -- (1.5em,0) node[midway,above] {\texttt{X}};
\draw[<-,very thick] (1,0) -- (1.5em,1.5em) node[midway,left] {\texttt{Y}};
\draw[<-,very thick] (0,0) -- (0,1.5em) node[midway,right] {\texttt{N}};
\end{tikzpicture}
\end{center}
\end{verbatim}
\begin{istgame}
\xtdistance{10mm}{20mm}
\istroot(0){1}
\istb{A}[al] \istb{B}[ar]{1,1} \endist
\istcntmarc{(cntm)(0-1)[green]+10mm..20mm+}
\istroot(1){cntm<135>2}+\cntmlevdist. \cntmlevdist+
\istb<missing> \istb{z}[l] \endist
\istroot(2){1-2}<\label distance=-4pt>45>{1}
\istb{Y}[l]{3-x,x} \istb{N}[r]{0,0} \endist
\end{istgame}

10.2 \texttt{\textbackslash istcntmarc} (arc version)

The second type macro \texttt{\textbackslash istcntmarc} is the arc version, which draws a background of two branches connected with an arc to represent a continuum of branches.

\begin{verbatim}
% \texttt{\textbackslash istcntmarc}
% syntax:
% \texttt{\textbackslash istcntmarc}[<grow keyval>](<coor1>)(<coor2>)\{<color,opt>\}{<num>}+levdist..sibdist+
% defaults:
% [south]()(0,0)[bend right]{.5}+8mm..3*8mm+
\end{verbatim}

The option \texttt{<num>} (0.5 by default) specifies the convex combination of the root and each of the background child nodes. The smaller \texttt{<num>} is, the closer is the connecting arc to the root.

\begin{verbatim}
% Example: \texttt{\textbackslash istcntmarc}
\begin{istgame}[scale=1.2]
\istcntmarc{ctm}
\istroot(0)(ctm){I}+\cntmlevdist.\cntmlevdist+
\istb{x}[r] \istb<missing> \endist
\xtdistance{10mm}{18mm}
\istroot(1){0-1}<\label distance=-3pt>120>{II}
\istb{Y}[al]{x,1-x} \istb{N}[ar]{0,0} \endist
\end{istgame}
\end{verbatim}

You can also change the line style and the curvature of the arc by options such as \texttt{very thick}, \texttt{red}, \texttt{dotted}, and \texttt{bend right=45}.

\begin{verbatim}
% Example: \texttt{\textbackslash istcntmarc}
\begin{istgame}[scale=1.2]
\istcntmarc{ctm}[bend right=45,very thick,red,dotted]{.65}
\istroot(0)(ctm){I}+\cntmlevdist.\cntmlevdist+
\istb{x}[r] \istb<missing> \endist
\xtdistance{10mm}{18mm}
\istroot(1){0-1}<\label distance=-3pt>120>{II}
\istb{Y}[al]{x,1-x} \istb{N}[ar]{0,0} \endist
\end{istgame}
\end{verbatim}

You can change the length of background branches connected by an arc in the same manner as used in \texttt{\textbackslash istcntm}.

10.3 Examples

In the following example, the base angle of the background isosceles triangle is 30°.
You can also rotate a tree with a continuum of branches. When it is necessary, you should use \[\texttt{[bend left]}\] to get a correct result as shown in the following example:
The \texttt{istgame} package also provides some auxiliary macros, which enable you to add players, action labels, payoffs, and others to your tree.

\texttt{\texttt{xtInfosetOwner}} is an auxiliary tool for putting the owner of an information set.
\texttt{\texttt{xtActionLabel}} is an auxiliary tool for putting an action label to a branch.
\texttt{\texttt{xtOwner}} is an auxiliary tool for putting the owner of a node (or a player).
\texttt{\texttt{xtPayoff}} is an auxiliary tool for putting payoffs.
\texttt{\texttt{xtNode}} is another tool for putting ‘something’ into a node. This ‘something’ could be the owner of a node.
\texttt{\texttt{xtNode*}} puts ‘something’ into a plain node.

Syntax:
\begin{verbatim}
\texttt{xtInfosetOwner(<from>)(<to>)(<owner>[<pos>,<node opt>])}
\texttt{xtActionLabel(<from>)(<to>)(<action>[<pos>,<node opt>])}
\texttt{xtOwner(<coor>)(<owner>[<pos>,<node opt>])}
\texttt{xtPayoff(<coor>)(<payoff>[<pos>,<node opt>])}
\texttt{xtNode[<opt>](<coor>)[<node style>,<node opt>]{<owner>}}
\texttt{xtNode*(<coor>){<owner>}}
\end{verbatim}

The auxiliary macros depend on the coordinates defined in the sequence of \texttt{\texttt{istroot}}--\texttt{\texttt{istb}}--\texttt{\texttt{endist}} and print their corresponding objects on or around the specified coordinates. So, in the above auxiliary macros, all the coordinates such as \texttt{(<from>)}, \texttt{(<to>)}, and \texttt{(<coor>)} are mandatory arguments.

\begin{verbatim}
% Example: auxiliary macros
\begin{istgame}[scale=2]
\istroot(0)[chance node]
\istb[dashed]{A}
\istb*
\istb*[blue,very thick]{B}[right,xshift=2pt,yshift=2pt,red]
\endist
\xtInfosetOwner(0)(0)(2.5pt)
\xtInfoset[thick](0-1)(0-2){1}
\xtInfoset[thick,red](0-2)(0-3){2}
\xtInfoset[dashed,bend right=25](0-1)(0-3)
%-----------------
\xtActionLabel(0)(0-3){\epsilon}[bl] % action label in math mode
\xtInfosetOwner(0-1)(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}
%-----------------
\istroot[east](a)(0)[chance node]
\istb \endist
\xtNode*(a-1){text}
\end{istgame}
\end{verbatim}
To specify the \texttt{[<pos>]} option for the above auxiliary macros (other than \texttt{\xtNode}), you can use the abbreviations \texttt{[l]}, \texttt{[r]}, \texttt{[a]}, \texttt{[b]}, \texttt{[al]}, \texttt{[ar]}, \texttt{[bl]}, and \texttt{[br]}. Each of these abbreviations works only when used alone without any other option keys within brackets. For example, the option \texttt{[b]} or \texttt{[below]} or \texttt{[below,xshift=5pt]} works, but not \texttt{[b,xshift=5pt]}.

\begin{center}
\begin{verbatim}
\begin{istgame}[font=\scriptsize]
\xtdistance{10mm}{20mm}
\istcmtm(ctm) \% period 1
\istroot(1)(ctm){1}+\cntmlevdist...10mm+
   \istb(x)\[r
   \istb<missing>
\endist
\istroot(A1)(1-1)<{\label distance=-3pt}135>{2}
   \istb(A)\[al]{x,1-x}
   \istb(R)\[ar]
\endist
\xtOwner(1){period 1}{xshift=30mm}
\istcmtm(2)(ctm){1-2} \% period 2
\istroot(2)(ctm){2}+\cntmlevdist...10mm+
   \istb(y)\[r
   \istb<missing>
\endist
\istroot(A2)(2-1)<{\label distance=-3pt}135>{1}
   \istb(A)\[al]{1-y,y}
   \istb(R)\[ar]
\endist
\xtOwner(2){period 2}{xshift=25mm}
\xtPayoff(A2-1){\left\{\makecell{\delta_{1}\gamma_{2}\\\delta_{2}\gamma_{2}}\right\} \cdots \right.}{\\left\{\makecell{\delta_{2}\gamma_{2}\\\delta_{2}\gamma_{2}}\right\} \cdots}
\istcmtm(3)(ctm){2-3} \% period 3
\istroot(3)(ctm){1}+\cntmlevdist...10mm+
   \istb(x)\[r
   \istb<missing>
\endist
\istroot(A3)(3-1)<{\label distance=-3pt}135>{2}
   \istb(A)\[al]{x,1-x}
   \istb(R)\[ar]{0,0}
\endist
\xtOwner(3){period 3}{xshift=20mm}
\xtPayoff(A3-1){\left\{\makecell{\delta_{1}^{2}\gamma_{2}\\\delta_{2}^{2}\gamma_{2}}\right\} \cdots \right.}{\\left\{\makecell{\delta_{2}^{2}\gamma_{2}\\\delta_{2}^{2}\gamma_{2}}\right\} \cdots}
\end{istgame}
\end{verbatim}
\end{center}

% Example:
\begin{itemize}
\item \begin{istgame}[font=\scriptsize]
\xtdistance{10mm}{20mm}
\istcmtm(ctm)
\istroot(1)(ctm){1}+\cntmlevdist...10mm+
   \istb(x)\[r
   \istb<missing>
\endist
\istroot(A1)(1-1)<{\label distance=-3pt}135>{2}
   \istb(A)\[al]{x,1-x}
   \istb(R)\[ar]
\endist
\xtOwner(1){period 1}{xshift=30mm}
\istcmtm(2)(ctm){1-2} \% period 2
\istroot(2)(ctm){2}+\cntmlevdist...10mm+
   \istb(y)\[r
   \istb<missing>
\endist
\istroot(A2)(2-1)<{\label distance=-3pt}135>{1}
   \istb(A)\[al]{1-y,y}
   \istb(R)\[ar]
\endist
\xtOwner(2){period 2}{xshift=25mm}
\xtPayoff(A2-1){\left\{\makecell{\delta_{1}\gamma_{2}\\\delta_{2}\gamma_{2}}\right\} \cdots \right.}{\\left\{\makecell{\delta_{2}\gamma_{2}\\\delta_{2}\gamma_{2}}\right\} \cdots}
\istcmtm(3)(ctm){2-3} \% period 3
\istroot(3)(ctm){1}+\cntmlevdist...10mm+
   \istb(x)\[r
   \istb<missing>
\endist
\istroot(A3)(3-1)<{\label distance=-3pt}135>{2}
   \istb(A)\[al]{x,1-x}
   \istb(R)\[ar]{0,0}
\endist
\xtOwner(3){period 3}{xshift=20mm}
\xtPayoff(A3-1){\left\{\makecell{\delta_{1}^{2}\gamma_{2}\\\delta_{2}^{2}\gamma_{2}}\right\} \cdots \right.}{\\left\{\makecell{\delta_{2}^{2}\gamma_{2}\\\delta_{2}^{2}\gamma_{2}}\right\} \cdots}
\end{istgame}
\end{itemize}
12 Representing subgames

Here is an example of a whole game tree.

\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial \textit{node}]\circlearrowleft(1) \istb(a)[a]{(2,2)}[l] \istb(D)[r] \endist
\istroot(1)(0-2)<135>\circlearrowleft(1)
\istb(L)[a] \istb(R)[ar] \endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>\circlearrowleft(2)
\istb(e1)[al]{(4,2)} \istb(r)[ar]{(1,1)} \endist
\istroot(3)(1-2)<45>\circlearrowleft(2)
\istb(e1)[al]{(3,2)} \istb(r)[ar]{(0,3)} \endist
\end{istgame}

\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial \textit{node}]\circlearrowleft(1) \istb(A)[a]{(2,2)}[l] \istb(D)[r] \endist
\istroot(1)(0-2)<135>\circlearrowleft(1)
\istb(L)[a] \istb(R)[ar] \endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>\circlearrowleft(2)
\istb(e1)[al]{(4,2)} \istb(r)[ar]{(1,1)} \endist
\istroot(3)(1-2)<45>\circlearrowleft(2)
\istb(e1)[al]{(3,2)} \istb(r)[ar]{(0,3)} \endist
\xtdistsubgameBox(1){(2-1)(3-2)}[solid,blue]
\end{istgame}

12.1 \texttt{xtSubgameBox(*)} (experimental)

\texttt{xtSubgameBox}

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

\begin{verbatim}
\% \texttt{xtSubgameBox(*)}
\% syntax:
\texttt{xtSubgame(*)}(<subroot coor>){(<coor1>) (<coor2>) ... }[<opt>]
\% options:
*: filled box
(<subroot coor>): the subroot of a subgame (mandatory)
{(coor1) (coor2) ...}: coordinates of terminal nodes (mandatory)
[<opt>]: color, line style
\% defaults:
(){}[rectangle,dashed,inner sep=20pt,rounded corners=15pt,black!50]
*(){}[rectangle,inner sep=20pt,rounded corners=15pt,red!20]
\end{verbatim}

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 dashed, \texttt{inner sep}=20pt, \texttt{rounded corners}=15pt, and black!50.
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]O>(1)
\istb(A)[a]@{(2,2)}[1] \istb(D)[r] \endist
\istroot(1)(0-2)<135>{1}
\istb(L)[a] \istb(R)[ar] \endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
\istb(ell)[a]@{(4,2)} \istb{r}[@{(1,1)}] \endist
\istroot(3)(1-2)<45>{2}
\istb(ell)[a]@{(3,2)} \istb{r}[@{(0,3)}] \endist
\xtSubgameBox*(1){(2-1)(3-2)}% [solid,blue,inner sep=10pt]
\end{istgame}

12.1.2 `\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)}[a] \istb(D)[b] \endist
\istroot(1)(0-2)<135>{1}
\istb(L)[a] \istb{R}[b] \endist
\xtdistance{10mm}{20mm}
\istroot(2)(1-1)<135>{2}
\istb(ell)[a]@{(4,2)} \istb{r}[@{(1,1)}] \endist
\istroot(3)(1-2)<45>{2}
\istb(ell)[a]@{(3,2)} \istb{r}[@{(0,3)}] \endist
\xtSubgameBox*(1){(2-1)(3-2)}% [inner xsep=30pt]
\end{istgame}

12.2 `\xtSubgameOval(*)` (experimental)

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

% `\xtSubgameOval(*)`
% syntax:
% `\xtSubgameOval(*)`{(subroot coor){{(coor1) (coor2) ... }}[<opt>]}% options:
% *: filled box
% (subroot coor): the subroot of a subgame (mandatory)
% {(coor1) (coor2) ...}: coordinates of terminal nodes (mandatory)
% [<opt>]: color, line style
% defaults:
% (){}{ellipse,dashed,inner sep=20pt,rounded corners=15pt,black!50}
% (*){}{ellipse,inner sep=20pt,rounded corners=15pt,red!20]
\xtSubgameOval(*) works just like \xtSubgameBox(*), except for one thing: an oval instead of a rectangle. \xtSubgameOval draws an oval (or an ellipse) to represent a subgame, and \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 \xtSubgameBox: first, changing inner sep, and secondly, shifting the subroot. In the following example, the options inner xsep=25pt and ([xshift=30pt]1) are used to change the size of the subgame oval. Its color is changed by blue!20.
12.3 More examples

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

Here is one more example, which shows how to draw a rectangle and circles to represent subgames.
13 Miscellany

13.1 Various branch types, directions, and lengths

What follows is an example of how to deal with various line types, directions, and lengths of branches.

\begin{istgame}
\istroot{(0)}[chance node]<[blue]{N}+10mm..30mm+
\istb{->[shorten >=.8pt]{p}[above left,red]}
\istb{->[shorten >=.8pt]{1-p}[ar]}
\endist
\istroot{(1)}(0-1)+15mm..30mm+
\istb[level distance=7mm,sibling distance=14mm>{L}[al]}
\istb*[->>>]{R}[ar]
\endist
\istroot{(2)}(0-2)+10mm..15mm+
\istb[->,dashed]{A}
\istb<grow=30,level distance=30mm>{a}[ar]{(1,-1)}[blue]
\istb<grow=-45>[blue,text=red,thick]{b}[ar]{(u_1,u_2)}[green]-90
\endist
\end{istgame}

13.2 Code reuse

13.2.1 Drawing subgames

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

\begin{istgame}
\istroot[-135](0)[initial node]<0>{1}
\istb[A][a]{(2,2)}[l]
\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][a]{(4,2)}[l]
\istb[r][ar]{(1,1)}
\endist
\istroot(3)(1-2)<45>{2}
\istb[ell][a]{(3,2)}[l]
\istb[r][ar]{(0,3)}
\endist
\end{istgame}
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.

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

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

\texttt{! Package pgf Error: No shape named 0-2 is known.}

This is because the first active \istroot refers to the undefined coordinate (0-2). In this case you need to change the line as:

\texttt{\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{verbatim}
% Example: a smaller subgame
\begin{istgame}
\xtdistance{15mm}{30mm}
\istroot[-135](0)[initial node]<0>{1}
% \istb(A)[a]{(2,2)}[1]
% \istb(0)[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
\istroot(3)(1-2)<45>{2}
  \istb(\ell)[al]{(3,2)}
  \istb(r)[ar]{(0,3)}
\endist
\end{istgame}
\end{verbatim}
It is not possible to make a subgame stay put where it appears in a whole game. (Replace \textbf{[black!5]} with \textcolor{white}{white} to make the rest of the game disappear in the following example.)

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

13.2.2 Backward induction

It is also easy to illustrate the procedure of backward induction, when you analyze an extensive game. By using \istb with options, you can change the color and the style of a line representing a branch and an arrow tip to it.

\begin{verbatim}
% Example: backward induction
\begin{istgame}
\xtdistance{15mm}{30mm}
istroot[-135]{(0)}\{initial node\}<0>{1}
istb[dashed]\{(A)\{a\}\{(2,2)\}\{1\}
istb[very thick,very blue]\{r\}{ar}\{(0,3)\}
\endist
\xtdistance{10mm}{20mm}
istroot{2}{(1-1)}<135>{2}
istb[very thick]\{(ell)\{al\}\{(4,2)\}
istb[thick,dotted]\{(r)\{ar\}\{(1,1)\}
\endist
\setistDecisionNodeStyle[black!5]
istroot{3}{(1-2)}<45>{2}
istb[thick,dotted]\{(ell)\{al\}\{(3,2)\}
istb[very thick,blue]\{r\}{ar}\{(0,3)\}
\endist
\end{istgame}
\end{verbatim}
13.2.3 Code reusability

Chen(2013)'s work is good to understand how drawing a game tree with TikZworks.

\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}

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

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

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.
14 Game tree examples

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

14.1 Simple examples

```latex
\begin{istgame}[-,>=stealth,shorten >=.8pt]
\xtdistance{15mm}{30mm}
\istroot(0)[initial node] (Child)
\istb{Good}[left]((0,2))
\istb{Bad}[right]
\endist
\istroot(1)(0-2)<30>(Parent)
\istb{Forgive}[left]((1,1))
\istb{Punish}[right]((-1,-1))
\endist
\end{istgame}
```

```latex
\begin{istgame}
\xtdistance{10mm}{30mm}
\istroot(0){1}
\istb(L)[al]
\istb(R)[ar]
\endist
\xtdistance{7mm}{15mm}
\istroot(1a)(0-1)
\istb(a)[al]{1,0}
\istb(b)[ar]{2,3}
\endist
\istroot(1b)(0-2)
\istb(c)[al]{0,1}
\istb(d)[ar]{-1,0}
\endist
\xtInfoset(1a)(1b){2}
\end{istgame}
```

```latex
\begin{istgame}
\xtShowEndPoints[oval node]
\xtdistance{10mm}{30mm}
\istroot(0){1}
\istb(L)[al]
\istb(R)[ar]
\endist
\xtdistance{7mm}{15mm}
\istroot(1a)(0-1)
\istb(a)[al]{1,0}
\istb(b)[ar]{2,3}
\endist
\istroot(1b)(0-2)
\istb(c)[al]{0,1}
\istb(d)[ar]{-1,0}
\endist
\xtInfoset[ bend left=30](1a)(1b){2}
\end{istgame}
```
14.2 A game tree with a strategic game
### 14.3 Larger game trees

```latex
\begin{istgame}
\xtShowEndPoints
\xtdistance{10mm}{40mm}
\istrroot{(0)}[\textit{initial node}]{1} \\istb \\istb \\istb \\endist
\xtdistance{10mm}{10mm}
\istrroot{(1)}(0-1) \\istb \\istb \\istb \\endist
\istrroot{(2)}(0-2) \\istb \\istb \\istb \\endist
\xtdistance{10mm}{20mm}
\istrroot{(1)}(0-3){2} \\istb \\istb \\endist
\xtdistance{10mm}{7mm}
\istrroot{(a)}(1-3){3} \\istb \\istb \\istb \\endist
\xtdistance{10mm}{14mm}
\istrroot{(b)}(2-3) \\istb \\istb \\endist
\istrroot{(c)}(3-1) \\istb \\istb \\endist
\istrroot{(d)}(3-2){3} \\istb \\istb \\endist
\xtInfoset{(1)}(2){2}
\xtInfoset{(2-3)}(3-1){3}
\xtInfoset{(3-2)}(3-2)
\end{istgame}
```

![Diagram of a larger game tree]

```latex
\begin{istgame}
\xtdistance{10mm}{40mm}
\istrroot{(0)}[\textit{initial node}]{1} \\istb \\istb \\istb \\istb \\endist
\xtdistance{10mm}{10mm}
\istrroot{(1)}(0-1) \\istb \\istb \\endist
\istrroot{(2)}(0-2) \\istb \\istb \\endist
\xtdistance{10mm}{20mm}
\istrroot{(3)}(0-3){2} \\istb \\istb \\endist
\xtdistance{10mm}{7mm}
\istrroot{(a)}(1-3){3} \\istb \\istb \\istb \\endist
\istrroot{(b)}(2-3) \\istb \\istb \\endist
\istrroot{(c)}(3-1) \\istb \\istb \\endist
\istrroot{(d)}(3-2){3} \\istb \\istb \\endist
\xtInfoset{(1)}(2){2}
\xtInfoset{(2-3)}(3-1){3}
\xtInfoset{(3-2)}(3-2)
\end{istgame}
```

![Diagram of a larger game tree]
14.4 Tic-tac-toe (sketch)

\begin{lstlisting}[scale=1.7,font=tiny]
\begin{istgame}
\begin{istroot}(0) \istb \istb \istb \istb \istb{\ldots} \istb \istb{8}[r]
\endist
\foreach \x in {1,...,4}
\begin{istroot}(a1)(0-\x) \istb{2}[l] \istb{\ldots} \istb{8}[r]
\endist
\istroot(a7)(0-7) \istb{1}[l] \istb{\ldots} \istb{8}[r]
\endist
\xtdistance{10mm}{2mm}
\istroot(A)(a1-3) \istb{2}[l] \istb{\ldots} \istb{8}[r] \endist
\istroot(B)(a1-8) \istb{2}[l] \istb{\ldots} \istb{8}[r] \endist
\istroot(C)(a7-8) \istb{1}[l] \istb{\ldots} \istb{8}[r] \endist
\istroot(By)(Bx-1) \istb{3}[l] \istb{\ldots} \istb{7}[r] \endist
\istroot(Bz)(By-1) \istb{4}[l] \istb{\ldots} \istb{7}[r] \endist
\xtPayoff{Bz-4}\vdots
\end{istgame}
\end{lstlisting}
14.5 Selten’s horse

\begin{istgame}
\xtdistance{5mm}{16mm}
\istroot{\text{-45}}{0} [\text{initial node}][1]
\istb{D}[r]
\istb<\text{grow=east, level distance=30mm}>[C][a]
\endist
\istroot{1}(0-1)+10mm..20mm+$
\istb{L}[a][3,3,2]
\istb{R}[a][0,0,0]
\endist
\istroot{-45}(a)(0-2){2}$
\istb{d}[r]
\istb<\text{grow=0, level distance=20mm}>[c][a][1,1,1][r]
\endist
\istroot{al}(a1)(a-1)+10mm..20mm+$
\istb{L}[a1][4,4,0]
\istb{R}[a1][0,0,1]
\endist
\xtInfoset{1}{\text{(a1)}}{3}
\end{istgame}

Figure 2: IGT 331.2

14.6 Centipede game

\begin{istgame}[scale=1.5]
\setistgrowdirection\text{(south east)}
\xtdistance{10mm}{20mm}
\istroot{0} [\text{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}{\text{(xshift=5mm)}2-2}$
\end{istgame}

\begin{istgame}
\xtInfoset{3}{\text{(xshift=5mm)}2-2}{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][\text{(100,100)}][r] \endist
\end{istgame}
14.7 Poker game

\begin{lstlisting}[language={pict2e}]
\begin{tikzpicture}[scale=1.7]
\t \node (root) {\textbf{N}};
\t \node (black) [below left of=root] {\textbf{Black}};
\t \node (red) [below right of=root] {\textbf{Red}};
\t \node (fold1) [below of=black] {Fold} [al] {1,-1};
\t \node (fold2) [below of=red] {Fold} [ar] {-1,1};
\t \node (raise1) [below of=fold1] {Raise} [ar] {1,1};
\t \node (raise2) [below of=fold2] {Raise} [al] {1,1};
\t \node (pass1) [below of=black] {Pass} [al] {1,-1};
\t \node (pass2) [below of=red] {Pass} [ar] {-1,1};
\t \node (meet1) [below of=pass1] {Meet} [ar] {2,-2};
\t \node (meet2) [below of=pass2] {Meet} [al] {2,-2};
\end{tikzpicture}
\end{lstlisting}
14.8 Poker game: growing to the right

\begin{lstgame}[scale=1.3]
\setistgrowdirection{0}  % default grow-direction is 'east' from now on
\xtdistance{15mm}{30mm}
\istroot(0)\{\text{chance node}\}\left\{\text{N}\}
\istb\{Black\}\{bl\}
\istb\{Red\}\{al\}
\endist
\xtdistance{15mm}{30mm}
\istroot(1-1)(0-1)\left\{\text{1}\}
\istb\{grow=-45, level distance=10mm\}\{\text{Fold}\}\{bl\}\{1,-1\}
\istb\{Raise\}\{al\}
\endist
\xtdistance{12mm}{24mm}
\istroot(1)\{\text{2}\}
\istb\{\text{Pass}\}\{bl\}\{1,-1\}
\istb\{\text{Meet}\}\{al\}\{-2,2\}
\enddist
\xtdistance{15mm}{30mm}
\istroot(1-2)(0-2)\left\{\text{1}\}
\istb\{grow=-45, level distance=10mm\}\{\text{Fold}\}\{bl\}\{-1,1\}[[\text{shift=5pt}below]
\istb\{Raise\}\{al\}
\endist
\xtdistance{12mm}{24mm}
\istroot(2)\{\text{2}\}
\istb\{\text{Pass}\}\{bl\}\{1,-1\}
\istb\{\text{Meet}\}\{al\}\{-2,2\}
\enddist
\xtInfosetO\{\text{(1-1-2)(1-2-2){2}}\}  \% \xtInfosetO
\xtActionLabel\{\text{(0)(0-1)}\}\{\frac{1}{2}\}\text{[ar]}
\xtActionLabel\{\text{(0)(0-2)}\}\{\frac{1}{2}\}\text{[br]}
\end{istgame}
% poker: growing east -- clockwise (swap version)
\begin{istgame}[scale=1.3]
\setistgrowdirection'{{0}\ % \setistgrowdirection'
\xtdistance{15mm}{30mm}
\istroot(0)[chance node]<left>{N}
  \istb(Black){al}
  \istb(Red){bl}
\endist
\xtdistance{15mm}{30mm}
\istroot(1-1)(0-1)<left>{1}
  \istb{grow=45,level distance=10mm}{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{grow=45,level distance=10mm}{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}
\xtActionLabel{0}(0-1){[\frac{1}{2}] [br]}
\xtActionLabel{0}(0-2){[\frac{1}{2}] [ar]}
\end{istgame}
14.9 Signaling games

% signaling game: IGT 337.1 (Osborne, 2004b)
\begin{istgame}
\setistgrowdirection{right}
% game start: choice of change
\xtdistance{20mm}{20mm}
\istroot(0) [chance node]<180>{Chance}
\istb{grow=north}{H}{1}
\istb{grow=south}{L}{1}
\endist
\istroot(H0)(0-1)<180>{F}
\istb{(p_1,E)}{a}
\endist
\istroot(L0)(0-2)<180>{F}
\istb{(p_1,E)}{a}
\endist
\xtdistance{10mm}{20mm}
% subtree after H is chosen
\istroot(H)(H0-1)
\istb{Buy}{br}
\istb{Refrain}[ar]{-E,0}
\endist
\istcntm(cntm)(H-1)+10mm..20mm+
\istroot(H1)(cntm)+10mm..8mm+
\istb{(p_2^H)}{b}
\istb<missing>
\istroot(CH)(H1-1)<(label distance=-4pt)135>{C}
\istb{Buy}{br}{p_1-E-c_H+p_2^H-c_H, 2H-p_1-p_2^H}
\istb{Refrain}[ar]{p_1-E-c_H, H-p_1}
\endist
% subtree after L is chosen
\istroot(L)(L0-1)
\istb{Buy}{br}
\istb{Refrain}[ar]{-E,0}
\endist
\istcntm(cntm)(L-1)+10mm..20mm+
\istroot(L1)(cntm)+10mm..8mm+
\istb<missing>
\istb(p_2^L){a}
\endist
\istroot(CL)(L1-2)<(label distance=-4pt)135>{C}
\istb{Buy}{br}{p_1-E-c_L+p_2^L-c_L, 2L-p_1-p_2^L}
\istb{Refrain}[ar]{p_1-E-c_L, L-p_1}
\endist
\xtInfoset(H)(L){C}[left]
\xtActionLabel(0)(0-1)\{\pi\}[r]
\xtActionLabel(0)(0-2)\{1-\pi\}[r]
\end{istgame}
\begin{istgame}[scale=1.3]
\xtdistance{20mm}{20mm}
\istroot(0)[chance node]{$c$}
\istb{grow=left}{$\frac{1}{2}$}[a]
\istb{grow=right}{$\frac{1}{2}$}[a]
\endist
\xtdistance{10mm}{20mm}
\istroot(1)(0-1)<180>{1}
\istb{grow=north}[a][l]
\istb{grow=south}[b][l]
\endist
\istroot(north)(a1)(1-1)
\istb{L}[al]{1,0}
\istb{R}[ar]{0,-1}
\endist
\istroot(b1)(1-2)
\istb{L}[al]{2,0}
\istb{R}[ar]{0,2}
\endist
\istroot(a2)(2-2)
\istb{L}[al]{3,0}
\istb{R}[ar]{0,3}
\endist
\istroot(north)(b2)(2-1)
\istb{L}[bl]{1,0}
\istb{R}[br]{0,1}
\endist
\xtInfoset(a1)(b2){2}
\xtInfoset(b1)(a2){2}
\end{istgame}
Version history

- v1.0 (2017/09/04) This version, to be submitted to CTAN
  - introduced `\cntmlevdist` and `\cntmsibdist`
  - introduced `\cntmdistance` (for internal use)
  - redefined related macros.
  - package document done

- v0.99g (2017/08/21)
  - redefined the `\istgame` environment to add `\>=stealth` as an option
  - added `text depth=.25ex` as an option to `\lstb` to get better alignment of action labels

- v0.99f (2017/08/02)
  - redefined `\xtInfoset0(')`
    * added `\begin{scope}[on background layer]...\end{scope}`

- v0.99e (2017/08/01)
  - introduced the prime(') versions to deal with `\grow` (this is the swap version in TikZ)
  - redefined `\istroot(')`
    - defined `\istrooto(') to replace `\istroot*`
      * `\istroot*` completely replaced by `\istrooto`
  - redefine `\istcntm`
    - defined `\istcntmarc`
      * `\istcntmarc*` completely replaced by `\istcntmarc`
  - redefined `\xtInfoset(')` and `\xtInfosetOwner(')`
    - introduced `\xtInfoset0(')`
      * `\xtInfoset*` completely replaced by `\xtInfoset0`
  - introduced `\xtgrowprime` introduced
  - redefined `\xtNode` accordingly

- v0.99d (2017/07/27)
  - defined `\xtSubgameBox(*)` and `\xtSubgameOval(*)` to express a subgame

- v0.99c (2017/07/20)
  - defined `\xtgrow` to deal with going ‘clockwise’ of branch arrangement
  - defined `\setistgrowkey` to change between `\grow` and `\grow'`
  - introduced `\setistgrowdirection'` by redefining `\setistgrowdirection`
    - updated ‘related macros’ accordingly

- v0.99b (2017/07/16)
  - renamed `zero node` as `plain node`
  - added `\setistPlainNodeStyle`
  - defined `\xtNode*` (for a terminal node text with a plain node)
  - redefined `\xt<...>NodeStyle` (now all arguments are optional)
  - renamed `\isthorizontalallabelshift` as `\istactionlabelxshift`, with default 1pt
  - renamed `\istverticalallabelshift` as `\istactionlabelyshift`, with default 2pt
• v0.99a (2017/03/10)
  - started rewriting package manual
  - introduced \istb. (for terminal nodes)
  - redefined \xtShowTerminalNodes
  - introduced \xtShowEndPoints and \xtHideEndPoints
  - redefined \xtInfoset*, with \xtdefaultinfosep added

• v0.9 (2017/02/13)
  - some internal macro names changed (\xtnode, \xttnode, \xtshowtnode)
  - added zero node

• v0.8 (2017/01/17)
  - macro names changed
    * \xtdistance: prefix ‘x’ changed to ‘xt’ meaning ‘extensive tree’
    * \xtInfoset(*), \xtInfosetOwner, \xtActionLabel, \xtOwner, \xtPayoff, \xtNode
    * \xtShowTerminalNodes, \xtHideTerminalNodes
    * \xtlevdist, \xtsibdist
  - \xtNode redefined
  - some internal macro names changed (including \xtpayff, \xtmove)
  - node styles redefined
  - \setist<Solid>NodeStyle added for <Various> node styles
  - \setistdefaultnodecolor, \setistdefaultnodebgcolor added

Acknowledgement

The basic idea of the istgame package came from Osborne’s egameps package and Chen (2013). A special thanks goes to Kangsoo Kim from KTUG for his great help in using expl3 to simplify the appearance and the usage of the macros.

References
