Drawing game theory trees using FOREST

Vojta Kovarik

January 7, 2019

To use these styles, use \input{GT_trees} – it imports all the necessary packages. This document provides examples of drawing pictures in TikZ + Forest, using some macros and styles that I have defined. Recommended order of reading is: if you only need one or two examples, and don't need to fine-tune them, then just copy the code from an example that you like. If you need more details, something doesn't behave as expected, or you start encountering errors, then read the section about TikZ/Forest basics, and the section relevant to your problem.

If this text doesn't solve your problem, consult the forest documentation and PGF/TikZ documentation (or draw your picture using some other tool). For drawing game trees without colored and/or triangle shaped player nodes, see Drawing Game Trees with Ti*k*Z.

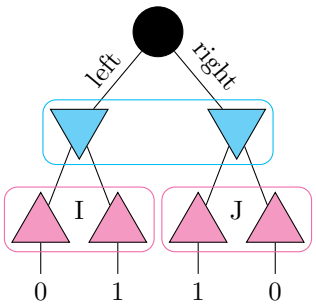


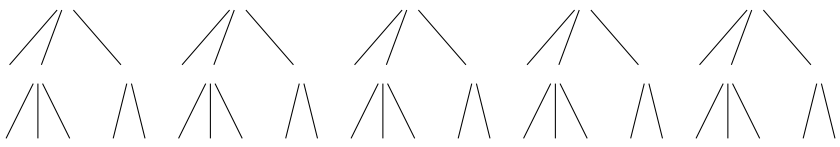
Figure 1: The purpose of this document is to allow a simple creation of game trees, such as this one, without having to learn the whole Ti*k*Z package.

Contents

1 Basic constructions using TikZ and Forest

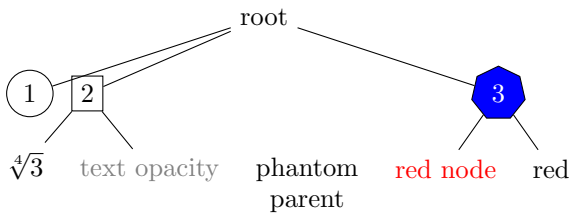
1.1 Forest syntax

The basic syntax using forest is just square brackets, commas serve to separate arguments but do not throw errors when there are too many of them. It is not sensitive to spaces, tabs and line ends. However, *empty lines throw errors*. See the latex code for examples.



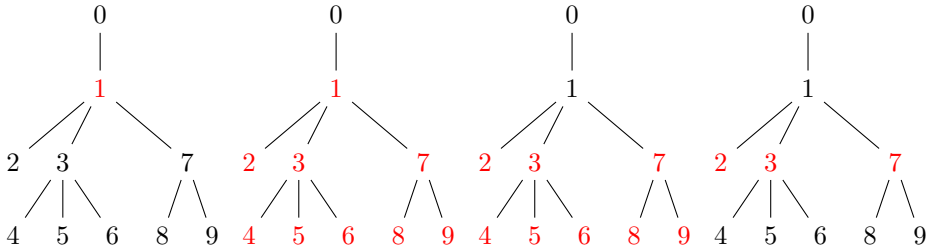
1.2 Formatting nodes

The first (optional) argument in each node is the text, other arguments influence formatting, and can be used in any order.



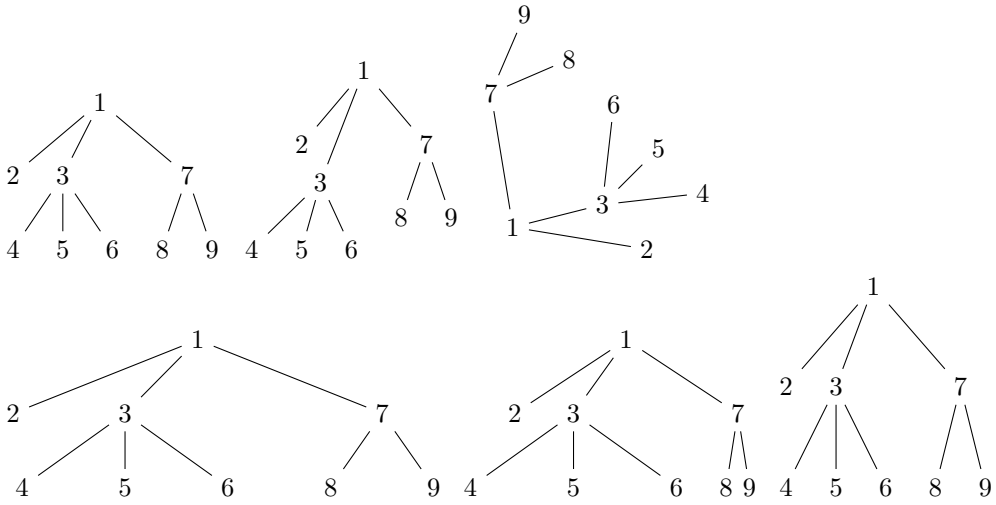
1.3 Formatting multiple nodes at once

Multiple nodes can be formatted at the same time with the help of the 'for tree/children/descendants' syntax:

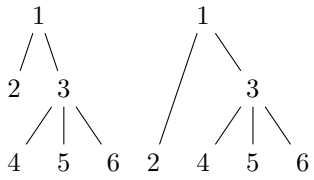


1.4 Changing the tree geometry

The position of a node/nodes can be adjusted using 'l', 'xshift', 'yshift' and 's sep' (and in many other ways). Growth direction can be adjusted using 'grow' argument (which takes either numeric angles or directions such as 'south east').



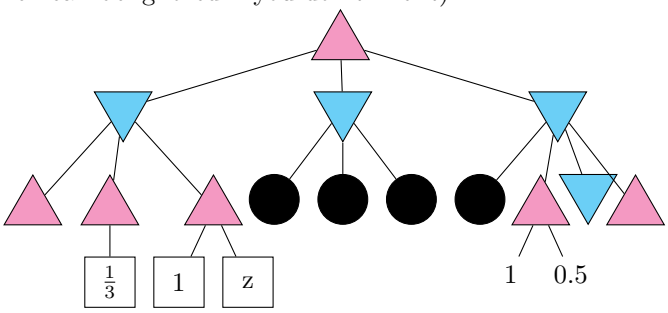
Use 'tier' enforce that specific nodes are vertically aligned:



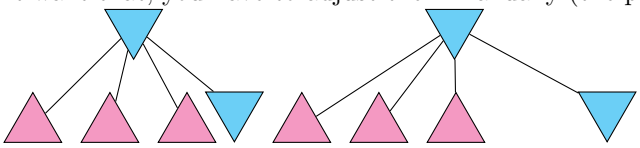
2 Custom game theory constructions

2.1 Node styles

The predefined game-theory nodes styles are ‘pl1’, ‘pl2’, ‘chance’ and ‘terminal’ (which can be ignored if you don’t like it):



Minimizer and maximizer nodes might end up too close to each other. If you don’t want that, you have to adjust them manually (the picture is an overkill):

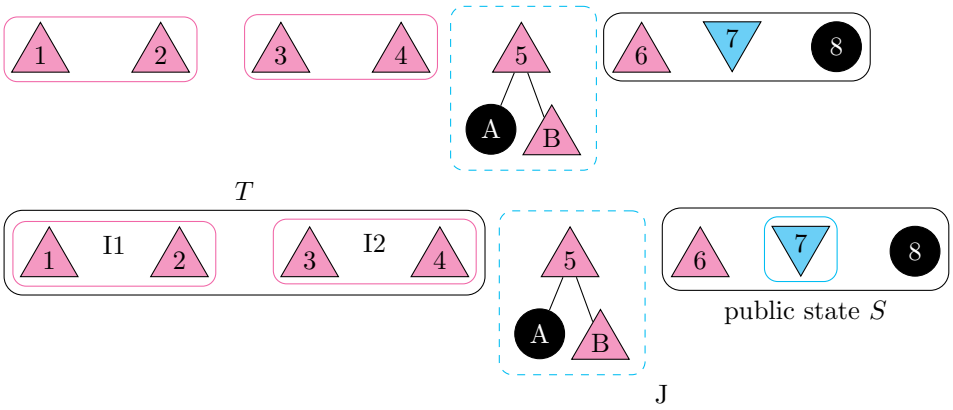


2.2 Information sets

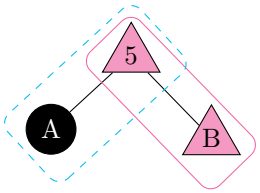
We show two ways of drawing information sets – rectangles and connecting by dashed lines.

2.2.1 Partitioning by rectangles

You can name nodes (and other objects) and create information sets using the “\node [style, fit=(nodeA)(nodeB)] ;” syntax. For triangle nodes, use (\corner{nodeA}) instead of (nodeA). The predefined styles are ‘infoset’, ‘opponent’ and ‘augmented’. Alternatively you can use ‘cl_infoset’ and ‘cl_opp_infoset’ to draw classical information sets which are better aligned with the corresponding triangles (but not with each other, if they appear on the same line).

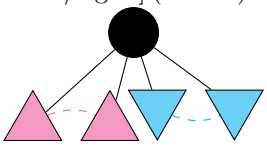


For infosets containing stuff from different levels, you can also use rotated infosets by adding 'rotate fit=α', where α is the rotation angle. The '\measureAngle' macro measures the angle automatically and saves it into '\angle'.

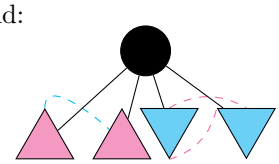


2.2.2 Connecting by dashed lines

You can also indicate infosets using curved dashed lines, using the '\draw[line_infoset, bend left/right] (nodeA) to (nodeB)' syntax. For player 2 color, use 'opp_line_infoset'.



You can also specify the exact angle, and where should the lines start and end:

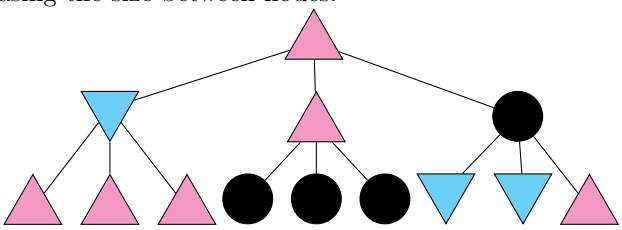


2.3 Multiple partitions in a single picture

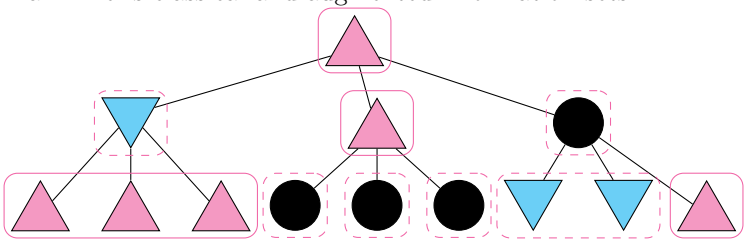
When multiple partitions are present in a single picture, you might have to increase the distances between nodes. This increases the size of the picture a lot, and there might often be a better solution (only drawing one player, drawing each player in a separate picture, drawing one player's infosets with dashed lines).

Adding infosets for different players (there should be a better solution than putting it all into a single picture – this is just for illustration):

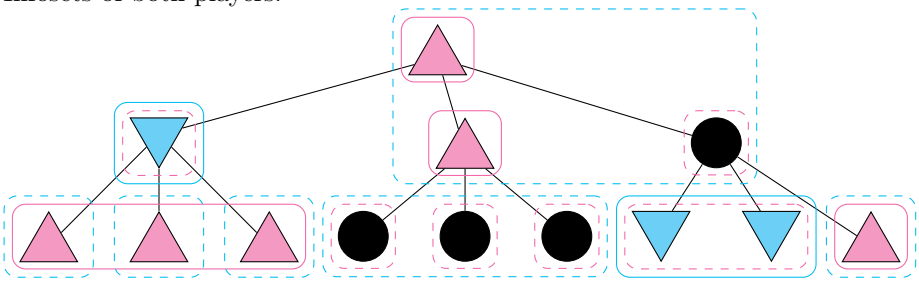
Basic example with no information sets, with commented-out lines for increasing the size between nodes:



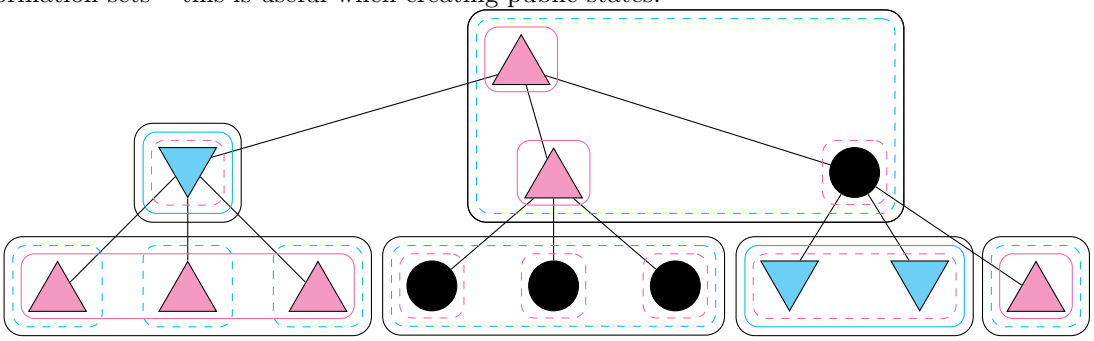
Maximizer's classical and augmented information sets.



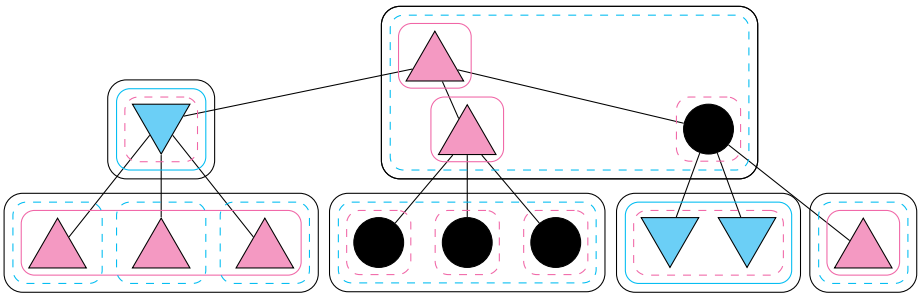
Infosets of both players.



The 'fit' function can be used not only with nodes as an input, but also with information sets – this is useful when creating public states.



The size can be often further reduced by fine-tuning 's sep', 'l' and 'l sep' for specific nodes (but it takes time):



3 Putting pictures into other text

3.1 Standard figure syntax

Pictures can be added directly into a specific place by just using the '\begin{forest} ... \end{forest}' syntax. You can also wrap it in the 'figure' environment as usual (Figure 2).

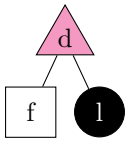
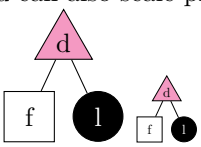


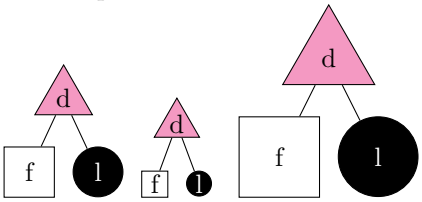
Figure 2: A sample figure. The label goes *after* the caption.

3.2 Scaling pictures

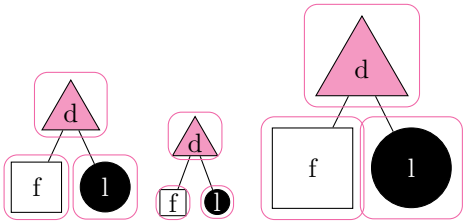
You can also scale pictures with scalebox:



There should be other ways of scaling the pictures, namely via adjusting the '\nodesize' parameter.



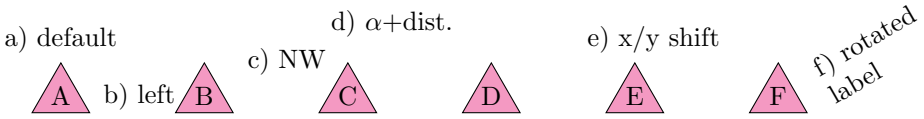
Unfortunately, this doesn't work well with infosets, and would (currently) require adjusting other parameters manually:



4 Naming nodes, infosets, and actions

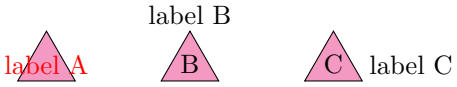
4.1 Labels

Labels can be specified with the 'label={[optional arguments] angle : label text}' syntax. Multi-line labels can be done with the 'align=...' environment (see F).



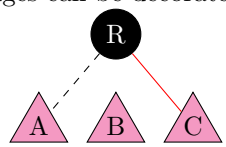
4.2 Extra nodes with text via TikZ

You can also add extra nodes with text manually, using TikZ:

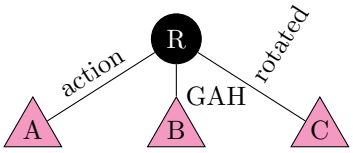


4.3 Decorating edges

Edges can be decorated with normal tikz syntax:



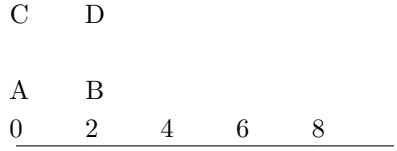
You can also put labels on top of edges:



5 Avoiding re-typing stuff: tikz loops, tree loops, latex newcommand

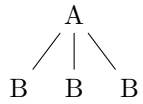
5.1 TikZ loops

TikZ has it's own for syntax – it iterates over tuples from a given set, and optionally has a counter



5.2 Creating multiple children on a single line

For a proper loop syntax in forest, see section 3.10 of forest manual. A lot of things can be dun just with 'repeat' and 'append' though:



Warning: Putting the tikz foreach cycles inside trees will often not produce the expected results.

5.3 Defining macros and new commands

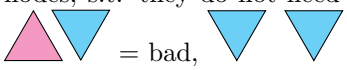
To avoid repetitive text, you can use latex's newcommand, via

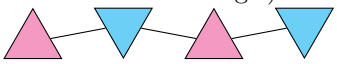
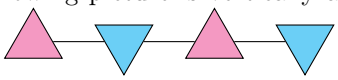
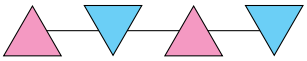
'\newcommand{\commandName}[numberOfParameters1]{ ... definition ... }',

Just wrap your repeated text in this with 0 parameters, and re-use it by '\commandName{input1}...{inputN}'.

6 Unresolved issues

- Figure out a way to scale pictures with infosets without changing the font size.
- Pre-defined prettier colors. The current combination is not terrible, but still fairly ugly.
- Automatically increase the distance between maximizer and minimizer nodes, s.t. they do not need specific treatment and manual adjustment.

 = bad, = good.

- Change anchors of triangle nodes (or do some other magic) s.t. the following picture is vertically aligned.  = bad,  = bad,  = better (but a text inside would still be aligned wrong).

7 Examples

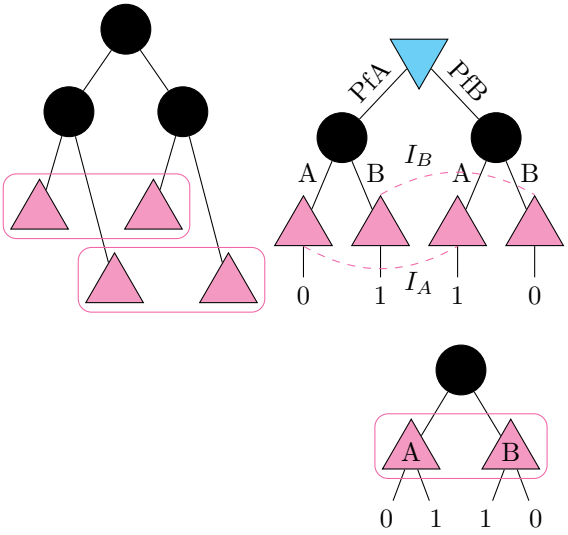


Figure 3: Caption. Label goes after, or it won't work.