The package tikz-inet

Marc de Falco

June 18, 2008

Abstract

The purpose of this package is to extend tikz with some simple macros in order to draw interaction nets.

Changelog

- **0.1** *initial release*

TODO

- add support for proofnets

1 Installation

Install this package in your ressource directory or the root of your document directory.

This package obviously needs tikz and pgf, but **this package needs the version, at least, 2.0 of them**.

2 Usage

2.1 Loading the package

The command to load the package is the usual

`\usepackage[<`*options*`>]{tikz-inet}`

where *options* are a comma separated list of keys ranging in

fancy Select a fancier style suitable for talks but likely to frighten referees

color= The global color used by the fancy style, must be xcolor compliant color

angle= The default orientation of cells, in degrees (default to 0, that is a cell pointing downward)

2.2 Cell nodes

The cells are displayed using the internal node system of tikz. A special shape *cellule* is inherited from the tikz shape isosceles triangle, thus sharing the original anchors.

A macro is defined to encapsulate the node creation.

`\inetcell[`*tikz display keys*`](`*node name*`){`*symbol*`}[`*angle*`]`

Every parameter is optional but the symbol.

symbol The symbol of the cell, that is the text displayed in the center of the cell. This text is not assumed to be in math mode.

node name This is the name of the node that will be used for referencing it or connecting wires. This name default to the symbol, which is a convenient way to make simple drawings, this can lead to some error as soon as the symbol is not simple, like a math text.

angle The angle of the principal port of the cell, defaults to 0 which means that the cell has a downward orientation with the principal port on the right.

Special values U, D, L and R are defined for up, down, left and right orientation of the cell.

tikz display keys This additional display keys are appended to the one used by the package and given to the node construction. As it is appended after, it can be used to redefine all display options.

A special key `arity` allows to define arity based anchors for ports. This is not really useful in most cases as the three predefined anchors are more likely to suit your need.

When using this macro a lot of parameters are given to tikz, one of them is the current cell display style. More on this can be found later.

2.2.1 Ports

Some anchors are named in order to reflect ports of the cell:

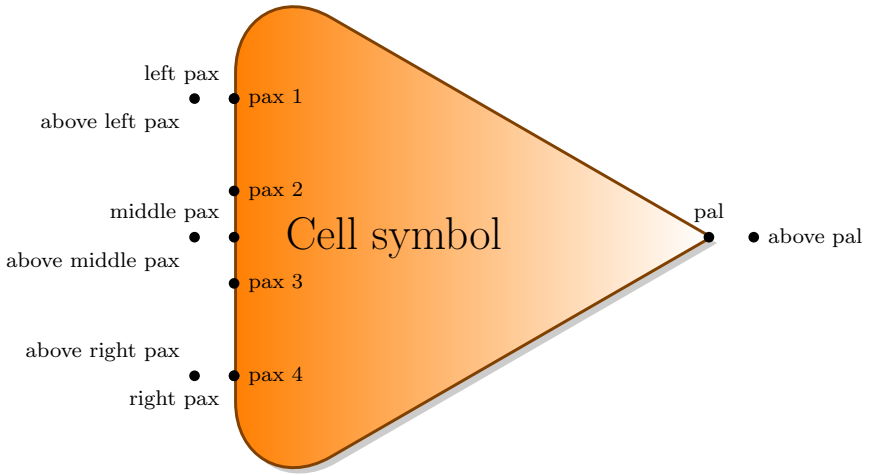
pal the apex of the triangle, where lies the principal port

middle pax an auxiliary port aligned with the principal port

left pax, right pax auxiliaries ports on each side of the opposite edge of the apex

pax n**,** $1 \leq n \leq arity$ numbered auxiliary ports associated with the arity key.

Each anchor has a sibling a bit off the cell, it allows nice curving when branching wires. To get this special anchor, just add `above` to the other ones: e.g. `above middle pax`



2.3 Wires

Wires macros help connecting ports by taking into account the current wire display style and dealing implicitly with the *above* anchors. The wires are displayed on a layer below the main one, so cells are always on top of layers. To change this you have to use `\pgfsetlayers`.

`\inetwire[`*tikz extra display keys*`](cell1.port1)(cell2.port2)`

The arguments are self-explanatory: this wire will connect the port `port1` of `cell1` with the port `port2` of `cell2`. It uses the current wire style.

`\inetloop[`*tikz extra display keys*`]`
draws a loop, that is a circle with the current wire style
`\inetwirecoords[`*tikz extra display keys*`](A)(B)`
draws a wire from node A to node B with the current wire style. It's useful to add extra wires non linking cell ports while using the current wire style.
`\inetwirefree[`*tikz extra display keys*`](cell.port)`
draws a wire from `port` of `cell` to `above port` of the same cell, with the current wire style. It allows fast declaration of free ports.

2.4 Boxes

Boxes uses the current box style when being displayed, they lie on a layer below the wire layer.
`\inetbox[`*tikz extra display keys*`]{`*space separated list of cells in the box*`}(`*box node name+*`)` display a box containing the given cells
`\inetprombox[`*tikz extra display keys*`]{`*space separated list of cells in the box*`}(`*promotion cell node name+*`)`: display a box and add a promotion cell below it, the name of the box is `bname` where `name` is the name of the promotion cell.

2.5 Other macros

`\inetnofancy`, `inetfancy`: hotswap the current display style
`\inetcellstyle`,`\inetwirestyle`,`\inetboxstyle`: get the currently used style for each type of drawing. **Don't renew this command to define your own style!**
`\inetcolor`: the current color for the fancy style, also passed as an argument of the non fancy style.
If you want to call a style you have to give an argument, unless you have redefined the fancy styles to not use it. For a cell like element you have to give the key `\inetcellstyle=\inetcolor`.

2.5.1 Tikz styles

This package define six styles :

- *cellstyle* and *fancycellstyle*
- *wirestyle* and *fancywirestyle*
- *boxstyle* and *fancyboxstyle*

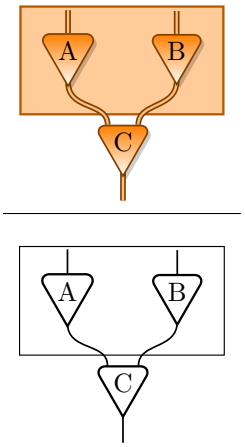
You can redefine this styles as for any other tikz styles with the command:
`\tikzset{`*the style name*`/.style={`*tikz display keys*`}}`

3 Examples

3.1 Basic

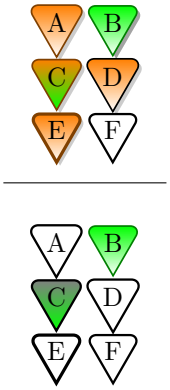
All examples are shown with the fancy key on the left.


```
\inetcell{A}
```



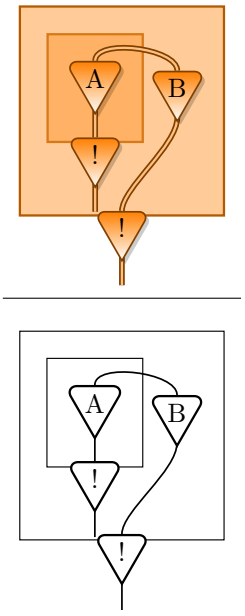

```
\matrix[row sep=0.5cm]{
    \inetcell{A} & & \inetcell{B} \\
    & \inetcell{C} & \\
};
\inetwirefree(A.middle pax)
\inetwirefree(B.middle pax)
\inetwirefree(C.pal)
\inetwire(A.pal)(C.right pax)
\inetwire(B.pal)(C.left pax)
\inetbox{(A) (B)}(b)
```

3.2 Style variations

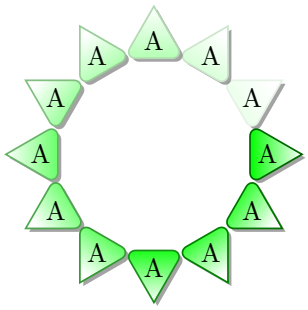


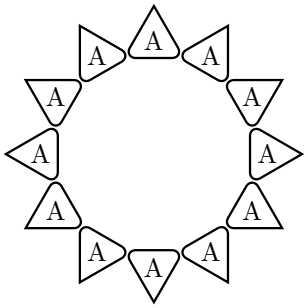

```
\matrix{
\inetcell{A} &
\inetcell[fancycellstyle=green]{B} \\
\inetcell[bottom color=green]{C} &
\inetcell[draw=black]{D} \\
\inetcell[very thick]{E} &
\inetnofancy \inetcell{F} \inetfancy \\
};
```

3.3 Special cases

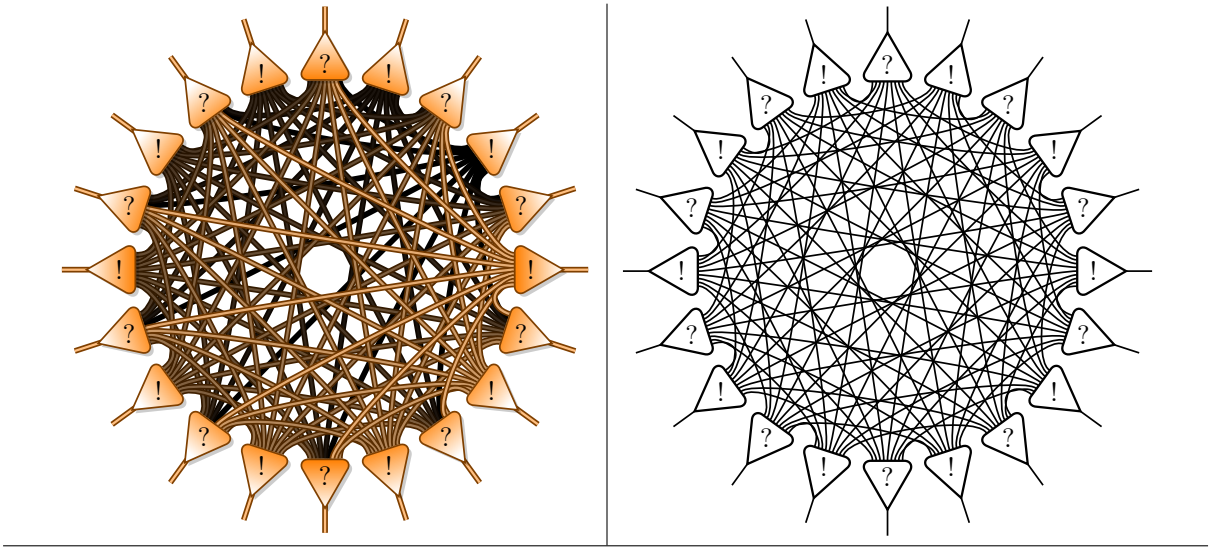



```
\inetcell{A}
\inetprombox{(A)}(pa)
\inetcell[at=(bpa.east),right=5pt]{B}
\inetwire(B.middle pax)(A.middle pax)
\inetprombox{(bpa)(pa)(B)}(p)
\inetwire(A.pal)(pa.middle pax)
\inetwirefree(pa.pal)
\inetwirefree(p.pal)
\inetwire(B.pal)(p.middle pax)
```






```
\newcount\angle
\foreach \x in {1,...,12} {
    \pgfmathsetcount{\angle}{360*\x/12+90}
    \inetcell[\inetcellstyle=green!\x0,
        at=(\the\angle-90:1.5cm)]
        (c\x){A}[\angle]
}
```



```
\newcount\angle
\newcount\order
\order=10
\newcount\arity
\pgfmathsetcount{\arity}{\order-1}
\foreach \x in {1,...,\order} {
    \foreach \y/\symbol in {0/!,1/?} {
        \pgfmathsetcount{\angle}
            {(180*(2*\x+\y))/\order+90}
        \inetcell[at=(\the\angle-90:\the\order*1.8ex),
            arity=\order-1](c\y\x){\symbol}[\angle]
        \inetwirefree(c\y\x.pal)
    }
}

\newcount\nextcell
\newcount\nextport
\newcount\depth
\foreach \x in {1,...,\order} {
    \foreach \y in {1,...,\arity} {
        \pgfmathsetcount{\nextcell}
            {mod(\x+\y-1,\order)+1}
        \pgfmathsetcount{\nextport}
            {\arity-\y+1}
        \pgfmathsetcount{\depth}{(\x-1)*100/\order}
        \inetwire[\inetwirestyle=\inetcolor!\the\depth!black]%
            (c0\x.pax \y)(c1\the\nextcell.pax \the\nextport)
    }
}
```