

# strid – A string diagrams generator

Samuel Mimram

May 8, 2008

## Contents

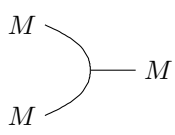
<b>1</b>	<b>Presentation of <i>strid</i></b>	<b>2</b>
1.1	A first example . . . . .	2
1.2	Visualizing your diagram . . . . .	4
1.3	Compiling . . . . .	4
<b>2</b>	<b>The operators</b>	<b>5</b>
2.1	Line: <code>line</code> . . . . .	5
2.2	Multiplication: <code>mult</code> . . . . .	5
2.3	Unit: <code>unit</code> . . . . .	5
2.4	Adjunction: <code>adj</code> . . . . .	5
2.5	Symmetry: <code>sym</code> . . . . .	6
2.6	Braiding: <code>braid</code> . . . . .	6
2.7	$m, n$ -ary box: <code>mboxn</code> . . . . .	6
2.8	Vertical box: <code>vbox</code> . . . . .	6
2.9	Region: <code>region</code> . . . . .	7
<b>3</b>	<b>Parameters of operators</b>	<b>7</b>
3.1	Labels . . . . .	7
3.1.1	Triangles: <code>triangle</code> / <code>t</code> . . . . .	8
3.1.2	Rectangles: <code>rectangle</code> / <code>r</code> . . . . .	9
3.2	Arrows . . . . .	9
<b>4</b>	<b>Configuration files</b>	<b>10</b>
<b>5</b>	<b>Examples</b>	<b>10</b>
5.1	Yang-Baxter equality for braids . . . . .	10
5.2	Hopf law for bialgebras . . . . .	11
5.3	Naturality condition for natural transformations between two lax functors between bicategories . . . . .	12

*strid* is a string diagrams generator for inclusion into  $\text{\LaTeX}$  files. It is entirely programmed in OCaml<sup>1</sup>. Feel free to drop me a line at [samuel.mimram@pps.jussieu.fr](mailto:samuel.mimram@pps.jussieu.fr) if you have some comments, bug reports or feature requests about it.


## 1 Presentation of *strid*

### 1.1 A first example

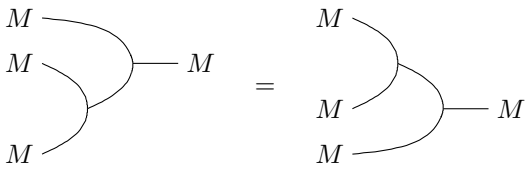
Suppose that  $(\mathcal{C}, \otimes, I)$  is a strict monoidal category. A *monoid* in  $\mathcal{C}$  is an object  $M$  of  $\mathcal{C}$  together with two maps  $\mu : M \otimes M \rightarrow M$ , called *multiplication*, and  $\eta : I \rightarrow M$ , called *unit*, respectively drawn as



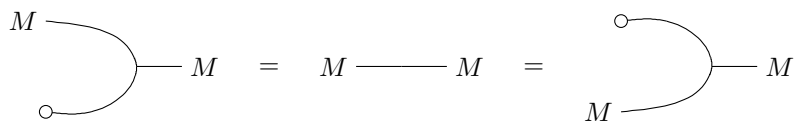
and



such that the equalities

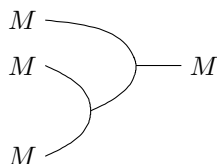


and



hold.

Let's have a look at how we typeset the left member of the associativity equation:



The *strid* code for this figure is

```
matrix {
text(r) [1,t=#M$#] \\
text(r) [1,t=#M$#]&&\mult(u1l,d1,r)&text(r) [1,t=#M$#]& \\
&\mult(u1,d1,) \\
text(r) [1,t=#M$#] \\
}
```

<sup>1</sup>OCaml can be downloaded at <http://caml.inria.fr/>.

Despite its apparent complexity, this code is very simple! Like every *strid* diagram, this code starts with “`matrix {`” and ends with “`}`”. Between those lines comes the actual description of the diagram. It is structured as a matrix whose columns are separated by “`&`” and whose lines are separated by “`\\`”.

The rightmost multiplication is typeset by

```
mult(ul1,d1,r)
```

Here, “`mult`” is the kind of the operator (a multiplication-shaped one) and its arguments specify that it should be linked to the relative positions  $(-2, 1)$  (`ul1` means up-left-left),  $(-1, -1)$  (`d1` means down-left) and  $(1, 0)$ . The order in which the links should be specified is indicated on the figure below:



As for other operators, links are specified inputs first and then outputs.

The labels are specified similarly by instructions like

```
text(r)[l,t=##M$#]
```

This creates a “`text`” operator from here to the relative position  $(1, 0)$ . The brackets “`[l,t=##M$#]`” are here to specify optional parameters related to this operator. The “`l`” indicates that we are going to add a label and the “`t=##M$#`” means that the label’s text should be “`$M$`”. The text between `#` is quoted uninterpreted.

Suppose that we have put the text of this figure in a file named `monoid_assoc_l.strid`. Compiling this file can be simply done by typing

```
strid monoid_assoc_l.strid
```

This generates a file `monoid_assoc_l.tex` which can be used in a  $\text{\LaTeX}$  file like:

```
\documentclass{article}

\usepackage{tikz}

\begin{document}
\input{monoid_assoc_l.tex}
\end{document}
```

You will need the `TikZ` package which can be downloaded at <http://sourceforge.net/projects/pgf/>.

Similarly, the right member of the equation is generated in a file `monoid_assoc_r.tex`. To have the equality sign between the two diagrams centered vertically you need to center the two diagrams. This can be done using the `\vcenter` and `\hbox`  $\text{\LaTeX}$  commands as shown in the following example:

```
\[
\vcenter{\hbox{\input{monoid_assoc_l.tex}}}
=
\vcenter{\hbox{\input{monoid_assoc_r.tex}}}
\]
```

## 1.2 Visualizing your diagram

Making a nice diagram is sometimes hard and L<sup>A</sup>T<sub>E</sub>X compilation of the diagrams usually takes some time to complete. If you want to quickly see the diagram generated by *strid* on a file `toto.strid`, type the command

```
strid -g toto.strid
```

This will open a window in which the output diagram is displayed, which is refreshed every time the file `toto.strid` is changed.

## 1.3 Compiling

Compiling diagrams can quickly become a tedious task : you have to run *strid* to generate a L<sup>A</sup>T<sub>E</sub>X file and then use `latex` or `pdflatex` to produce the final document (which can take long to compile). If you want to speed up the compilation of the document, you can have *strid* generate directly pdf files by typing

```
strid --pdf toto.strid
```

The resulting `toto.pdf` file can then be integrated in a L<sup>A</sup>T<sub>E</sub>X document as follows.

```
\documentclass{article}

\usepackage{graphics}

\begin{document}
\[\includegraphics{toto.pdf}\]
\end{document}
```

In order to have the pdf files automatically generated from the strid files, a `Makefile` file can be used, containing

```
STRIDFILES=$(wildcard *.strid)
STRIDPDF=$(STRIDFILES:.strid=.pdf)

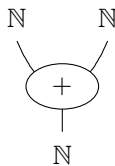
pdf: $(STRIDPDF)

%.pdf: %.strid
    $(STRID) --pdf $<
```

The pdf files corresponding to the strid files of the current directory can then be generated by simply typing

```
make
```

Be careful, if you use macros in your strid files those won't be known when generating the pdf. The solution is to put your macros in a `macros.sty` file (or whatever name.sty) and include those in the tex file used to produce the pdf by using the `--latex-preamble` command of *strid*. For example,



is typeset in a file `add.strid` containing

```

matrix {
text(d) [1,t=#$\N$#]&&text(d) [1,t=#$\N$#]\\
\\
&mult(u1,ur,d) [1,t=#$+$#]&\\
\\
&text(u) [1,t=#$\N$#]&\\
}

```

and is compiled with

```
strid --pdf --latex-preamble "\usepackage{macros}" add.strid
```

where the `macros.sty` file contains

```

\usepackage{amfonts}
\newcommand{\N}{\mathbb{N}}

```

The `Makefile` above can obviously be modified in order to cope with such situations.

## 2 The operators

### 2.1 Line: line



### 2.2 Multiplication: mult



### 2.3 Unit: unit



### 2.4 Adjunction: adj



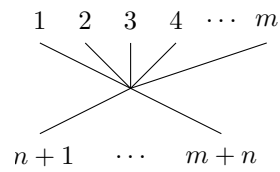
## 2.5 Symmetry: sym



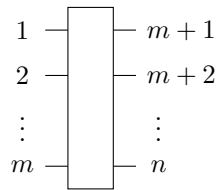
## 2.6 Braiding: braid



## 2.7 $m, n$ -ary box: $mboxn$

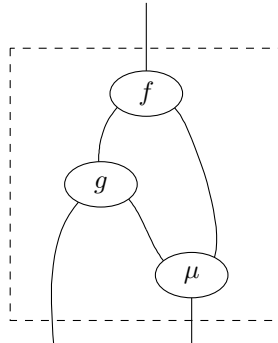


## 2.8 Vertical box: vbox



## 2.9 Region: region

Regions can be delimited:



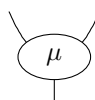
is typeset by

```
matrix {
\\
region(6d6r)&&&&&&\\
&&&mult(dl,dr,uu)[1,t=#f$#]\\
\\
&&mult(4dl,dr,u)[1,t=#g$#]\\
\\
&&&mult(ul,uuu,dd)[1,t=#μ$#]\\
\\
}
```

## 3 Parameters of operators

### 3.1 Labels

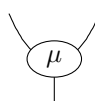
Labels can be added to operators. For example the diagram



can be typeset by

```
matrix {
\\
&mult(ul,ur,d)[1,t=#μ$#]&\\
\\
}
```

If you don't like the size of the ellipse surrounding the label, this can of course be changed. For example,



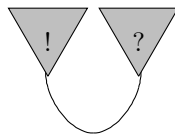
can be typeset by

```
matrix {
\\
&mult(u1,ur,d) [1,t=#$\mu$#,w=0.6,h=0.4]&\\
\\
}
```

Various shapes are available for labels:

### 3.1.1 Triangles: `triangle` / `t`

For example,

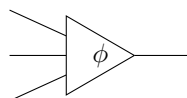


can be typeset by

```
matrix {
\\
&unit(d) [1,t=#$!$#,s=triangle,d=d,c=lightgray]&&
unit(d) [1,t=#$?$#,s=triangle,d=d,c=lightgray]&\\
\\
&&arc(u1,ur)&\\
}
```

Here, the `s` parameter is the *shape*, the `d` parameter is the *direction* of the triangle (here it is pointing down) and the `c` parameter specifies the *color* of the triangle. There are shortcuts for the shapes, for example you can type `s=t` instead of `s=triangle`, which is more concise but less readable.

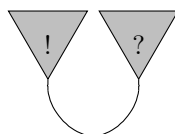
There is a particular case where the direction of the triangle can be guessed and you don't need to specify the direction of the triangle: when there is only one output port. The `operad` operator does precisely this.



can be typeset by

```
matrix {
\\
&&operad(d2l,2l,u2l,2r) [1,t=#$\phi$#]&&\\
\\
}
```

It also provides minor improvements of the drawing for example, the output wire is exactly starting at the vertex of the triangle :

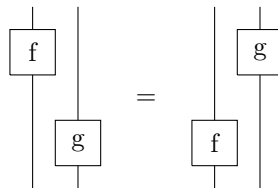


is typeset by



```
matrix {
\\
&operad(d)[1,t=#!$#,c=lightgray]&&
operad(d)[1,t=#?$#$,c=lightgray]&\\
\\
&&arc(ul,ur)&\\
}
```

### 3.1.2 Rectangles: rectangle / r



The left member is typeset by

```
matrix {
\\
1box1(u,3d)[1,t=#f#,s=rectangle]\\
\\
&1box1(3u,d)[1,t=#g#,s=rectangle]\\
\\
}
```

## 3.2 Arrows

Lines can be oriented using the **a** attribute. For example,



can be typeset by

```
matrix {
\\
&mult(ul,dl,r)[a]&\\
\\
}
```

To specify that the direction should be backwards use the **d=b** subattribute. For example,



can by typeset by

```
matrix {
\\
```

```
&mult(u1,d1,r)[a,d=b]&\
\\
}
```

The position of the arrow can be changed by setting the `t` parameter which is a float between 0. and 1.. For example, the arrow can be put at the end of a line by using the `t=1.` parameter :



is typeset by

```
matrix {
line(2r)[a,t=1.]&&\
}
```

## 4 Configuration files

All parameters can be saved in a configuration file named `strid.conf`. To generate a configuration file, type

```
strid --dump-conf
```

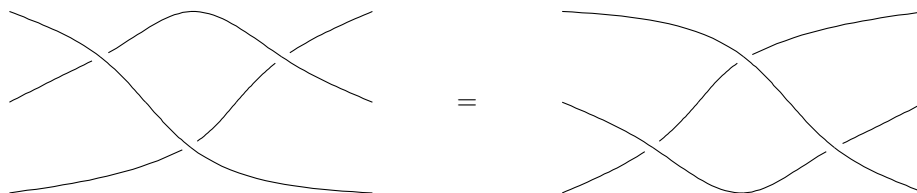
You can then edit `strid.conf`.

Some of the options that can be set are:

- `line_width`: default width of a line
- `label_width`: default width of a label
- `label_height`: default height of a label
- `no_tex_environment`: do not output `\begin{tikz}` and `\end{tikz}`
- `scaling_factor`: scale the diagrams
- `label_triangle_height`: default height of a triangular label
- `label_rectangle_width`: default width of a rectangular label
- `label_rectangle_height`: default height of a rectangular label
- `interpolation`: interpolation method for drawing lines (possible values are `cspline` and `linear`)
- `small_circle_ray`: ray of small circles (used to tweak the drawing of multiplications)

## 5 Examples

### 5.1 Yang-Baxter equality for braids



Left member is typeset by

```

matrix {
\\
&&braid(ull,dll,urr,dr)&&&braid(ull,dl,urr,drr)&&\\
\\
&&&braid(ul,d4l,ur,d4r)\\
\\
}

```

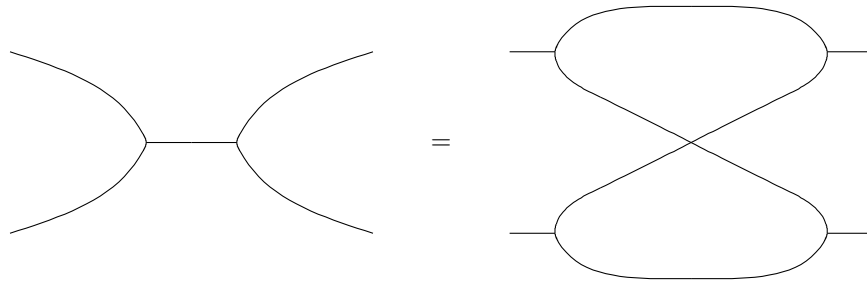
and right member by

```

matrix {
\\
&&&braid(u4l,dl,u4r,dr)\\
\\
&&braid(ull,dll,ur,drr)&&&braid(ul,dll,urr,drr)&&\\
\\
}

```

## 5.2 Hopf law for bialgebras



Left member is typeset by

```

matrix{
\\
\\
\\
&&&mult(uu3l,dd3l,r)&&mult(uu3r,dd3r,l)&&&\\
\\
\\
\\
}

```

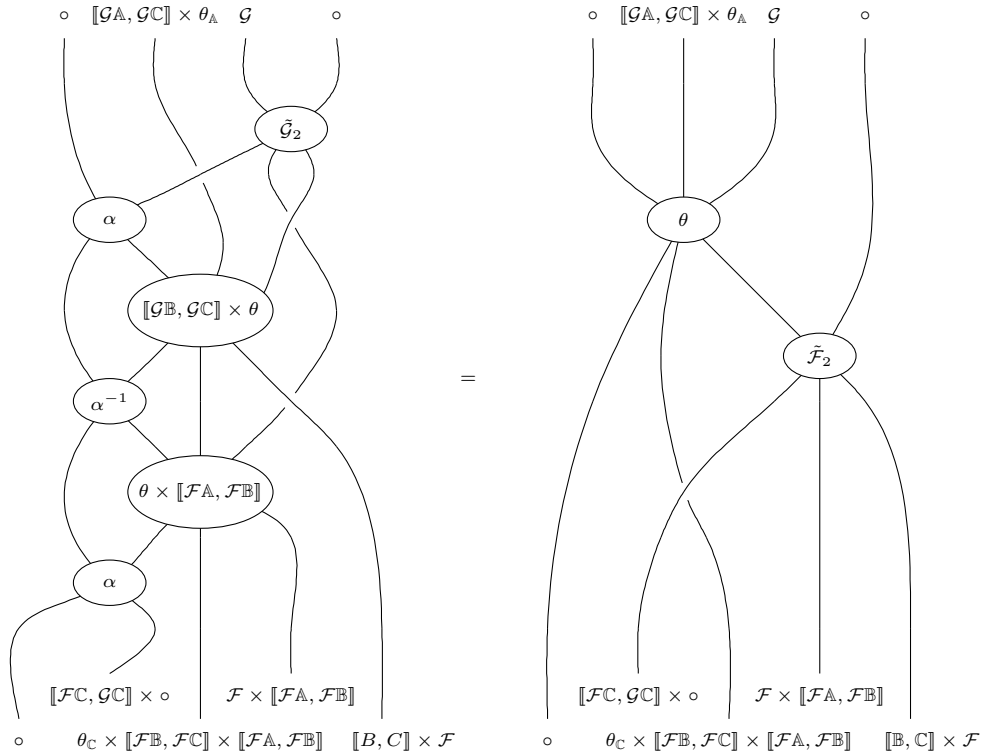
and right member by

```

matrix{
\\
&mult(u3r,dr,l)&&&&mult(u3l,dl,r)&&\\
\\
&&&sym(ull,dll,urr,drr)\\
\\
&mult(ur,d3r,l)&&&&mult(ul,d3l,r)&&\\
\\
}

```

### 5.3 Naturality condition for natural transformations between two lax functors between bicategories



The code for the left-hand side of the equation is

```

matrix {
\\
&text(u)[1,t=#$\circ$#]
&& text(u)[1,t=#$[\![\mathcal{G}\backslash A,\mathcal{G}\backslash C]\!] \times \theta\_A$#]
&& text(u)[1,t=#$\mathcal{G}$#]
&& text(u)[1,t=#$\circ$#]&\\
\\
&line(u)&&line(u)&&line(u)&&line(u)& \\
&&&&&&2box3(u1,ur,0.5dl,d0.5r)[1,t=#$\tilde{\mathcal{G}}\_2$#]&&\\
&&&&&braid(0.5ur,2u1,1.5d0.5r,0.5dl)&&&&\\
&&sym(3u1,0.5ur,2dl,dr)[1,t=#$\alpha$#]&&&&braid(u0.5r,u0.5l,2dr,2dl)&&\\
\\
&&&&3box3(u1,1.5u0.5r,r,d1,2d,dr)[1,t=#$[\![\mathcal{G}\backslash B,\mathcal{G}\backslash C]\!] \times \theta\_B,w=1.6,h=0.8]&&&&\\
\\
&&sym(2u1,ur,2dl,dr)[1,t=#$\alpha^{-1}$#]&&&&braid(u1,2ur,d1,dr)&&\\
\\
&&&&3box3(u1,2u,ur,d1,d,drr)[1,t=#$\theta \times [\![\mathcal{F}\backslash A,\mathcal{F}\backslash B]\!]$#,w=1.6,h=0.8]&&&&\\
\\
&&sym(2u1,ur,d1l,dr)[1,t=#$\alpha$#] && &&&&\\
line(,2d)&&&&line(2u,2d)&&line(2u,d)&&line(4u1,2d)\\
&&line(,ur)&&&&\\
\\
&&
text(u)[1,t=#$[\![\mathcal{F}\backslash C,\mathcal{F}\backslash C]\!] \times \circ$#]
&& &&
text(u)[1,t=#$\mathcal{F} \times [\![\mathcal{F}\backslash A,\mathcal{F}\backslash B]\!]$#]
&& \\
text(u)[1,t=#$\circ$#]
&& &&
text(u)[1,t=#$\theta\_C \times [\![\mathcal{F}\backslash B,\mathcal{F}\backslash C]\!] \times [\![\mathcal{F}\backslash A,\mathcal{F}\backslash B]\!]$#]
&& &&
text(u)[1,t=#$\sqquad [\![B,C]\!] \times \mathcal{F}$#]
\\
}

```

and the code for the right-hand side is

```
matrix {
  \\\
  &text(u)[1,t=#$\circ$#]&&text(u)[1,t=#$[\!\[\mathcal{G}\backslash A,\mathcal{G}\backslash C\backslash!]\ \times\ \theta_A$#]
  &&text(u)[1,t=#$\mathcal{G}$#]&&text(u)[1,t=#$\circ$#]&\
  \\\
  &line(u,d)&&line(u,d)&&line(u,d)&&line(u,4d)&\ \
  \\\
  &&&3box3(2u2l,2u,2u2r,4d2l,3d0.5l,dr)[1,t=#$\theta$#]&&&&\
  \\\
  &&&&2box3(2u2l,2ur,dl,d,dr)[1,t=#$\tilde{\mathcal{F}}_2$#]&&\
  \\\
  &&&braid(2u2r,3u0.5l,d0.5r,d0.5l)&&&&\
  \\\
  \\\
  line(5ur,dd)&&line(2u0.5r,d)&&line(2u0.5l,dd)&&line(5u,d)&&line(5ul,dd)\
  \\\
  &&
  text(u)[1,t=#$[\!\[\mathcal{F}\backslash C,\mathcal{G}\backslash C\backslash!]\ \times\ \circ$#]
  && &&
  text(u)[1,t=#$\mathcal{F}\ \times\ [\!\[\mathcal{F}\backslash A,\mathcal{F}\backslash B\backslash!]\ $#]
  && \\\
  text(u)[1,t=#$\circ$#]
  && &&
  text(u)[1,t=#$\theta_C\ \times\ [\!\[\mathcal{F}\backslash B,\mathcal{F}\backslash C\backslash!]\
  \times\ [\!\[\mathcal{F}\backslash A,\mathcal{F}\backslash B\backslash!]\ $#]
  && &&
  text(u)[1,t=#$\quad [\!\[\mathcal{B},\mathcal{C}\backslash!]\ \times\ \mathcal{F}$#]
  \\\
}
```