Documentation for `llconnex.sty` Version 0.99

J.M. Egger

June 14, 2007

Contents

1 Introduction

The purpose of `llconnex` is to define macros for the connectives of linear logic. The problem is that different authors use different symbols for these connectives, and that we wish to accommodate as many different tastes as possible.

My solution to this problem is two-fold. Firstly, to adopt the philosophy of Paul Taylor: that the source-code of the body of any TEX document should reflect what the author *means*, rather than the notation which the author happens to be using. Secondly, to equip `llconnex` with options which take over the job of translating concepts into symbols in accordance with the user's preferred convention.

Thus the correct usage of `llconnex` is to include

$$\texttt{\textbackslash usepackage[}\mathit{opt}_1, \ldots, \mathit{opt}_n\texttt{]\{llconnex\}}$$

in the preamble of one's document.

At present, there exist three sets of options: one to control the denotations of conjunctions and disjunctions; one to control the denotations of negations; and one to control the denotations of implications. Future versions of `llconnex` may incorporate yet more.

It is possible to use `llconnex` without options—for example, the present document was prepared in this way—but this is only recommended if, as in the present document, the user needs to refer to symbols independently of the concepts they are intended to represent.

1.1 System Requirements

`llconnex` makes frequent use of `stmaryrd.sty` which appears to have become a standard tool. It is freely available at . . .

2 Conjunctions and Disjunctions

The macros assigned to each of the (binary) conjunctions and disjunctions of linear logic are as follows:

multiplicative conjunction (*alias*: tensor, *tenseur*)	$\mapsto$	`\tens`
multiplicative disjunction (*alias*: par)	$\mapsto$	`\parr`
additive conjunction (*alias*: with, *avec*)	$\mapsto$	`\with`, `\avec`
additive disjunction (*alias*: plus)	$\mapsto$	`\plus`

The default macros for the units corresponding to each binary are

`\uftens`	(unit for `\tens`)	`\uptens`	(unite pour `\tens`)
`\ufparr`	(unit for `\parr`)	`\upparr`	(unite pour `\parr`)
`\ufwith`	(unit for `\with`)	`\upavec`	(unite pour `\avec`)
`\ufplus`	(unit for `\plus`)	`\upplus`	(unite pour `\plus`)

[These can be abbreviated using option `uns`—see below.]

Thus the users of `llconnex` are encouraged to type, for example, `x \parr \uftens` (... or, `x \parr \e`)

whenever they mean "x par (the unit for tensor)"—irregardless of which symbols they happen to be using to denote "par" or "the unit for tensor".

2.1 The orthodox option: `jyg`

Linear logic was, of course, invented by J.-Y. Girard [3]; his notation is as follows:

`x \tens y`	$\mapsto$	$x \otimes y$	`\uftens`	$\mapsto$	1
`x \parr y`	$\mapsto$	$x \parr y$	`\ufparr`	$\mapsto$	$\bot$
`x \with y`	$\mapsto$	$x \,\&\, y$	`\ufwith`	$\mapsto$	$\top$
`x \plus y`	$\mapsto$	$x \oplus y$	`\ufplus`	$\mapsto$	0

2.2 A common heterodox option: `cns`

Many category theorists object to Girard's notation for a variety of reasons. Commonly cited objections include that:

1. the symbol $\parr$ is ugly and difficult to reproduce (both on the blackboard and in TeX); and,

2. category theorists "always" use $\times$ to denote product (*i.e.*, with), and $+$ for coproduct (*i.e.*, plus).

[Both of these are easily refutable, but that is not the point.]

The notation used by M. Barr (who first defined the notion of $*$-*autonomous category* [1], which is often used to model MLL), J.R.B. Cockett and R.A.G. Seely (who are jointly responsible for the definition of *linear distributive* category [2], which can be used to model MLL-without-negation) is as follows:

`x \tens y`	$\mapsto$	$x \otimes y$	`\uftens`	$\mapsto$	$\top$
`x \parr y`	$\mapsto$	$x \oplus y$	`\ufparr`	$\mapsto$	$\bot$
`x \with y`	$\mapsto$	$x \times y$	`\ufwith`	$\mapsto$	1
`x \plus y`	$\mapsto$	$x + y$	`\ufplus`	$\mapsto$	0

2.3 A quantale theoretic option: `cjm`

Prior to J.-Y. Girard's invention of linear logic, C.J. Mulvey introduced quantale theory [4]. Quantales only model a fragment of linear logic, but it is an interesting fragment—in particular, they capture Girard's *phase space semantics*, [5].

Thus it seems only fair for `llconnex` to include a quantale-theoretic option:

`x \tens y`	$\mapsto$	$x \,\&\, y$	`\uftens`	$\mapsto$	e
`x \parr y`	$\mapsto$	$x \parr y$	`\ufparr`	$\mapsto$	d
`x \with y`	$\mapsto$	$x \wedge y$	`\ufwith`	$\mapsto$	1
`x \plus y`	$\mapsto$	$x \vee y$	`\ufplus`	$\mapsto$	0

2.4 A very strange option indeed: `jme`

To make matters worse, I entered the scene. I objected to all but the quantale-theoretic option as a result of the following inferences:

$$\dfrac{\dfrac{(\text{The symbol } \otimes \text{ originiated in linear algebra}) \qquad (\text{In } \mathbf{Vec}_{fd},\ \textit{tensor} \text{ and } \textit{par} \text{ coincide})}{(\text{The original meaning of the symbol } \otimes \text{ is that } \textit{tensor} \text{ and } \textit{par} \text{ coincide})}}{(\text{The symbol } \otimes \text{ should only be used when } \textit{tensor} \text{ and } \textit{par} \text{ coincide})}$$

$$\dfrac{\dfrac{[\text{The symbol } \oplus \text{ originiated in linear algebra}] \qquad [\text{In } \mathbf{Vec}_{fd},\ \textit{with} \text{ and } \textit{plus} \text{ coincide}]}{[\text{The original meaning of the symbol } \oplus \text{ is that } \textit{with} \text{ and } \textit{plus} \text{ coincide}]}}{[\text{The symbol } \oplus \text{ should only be used when } \textit{with} \text{ and } \textit{plus} \text{ coincide}]}$$

Eventually I hit upon the idea of breaking each of the symbols $\otimes$ and $\oplus$ into a conjunctive and a disjunctive part, and was thus led to the following system of notation:

`x \tens y`	$\mapsto$	$x \mathbin{\cap\!\!\!\!\times} y$	`\uftens`	$\mapsto$	e
`x \parr y`	$\mapsto$	$x \mathbin{\cup\!\!\!\!\times} y$	`\ufparr`	$\mapsto$	d
`x \with y`	$\mapsto$	$x \mathbin{\cap\!\!\!\!+} y$	`\ufwith`	$\mapsto$	t
`x \plus y`	$\mapsto$	$x \uplus y$	`\ufplus`	$\mapsto$	o

[Unfortunately, it is more difficult to break apart the traditional symbols for the units of $\otimes$ and $\oplus$ (k and 0) in a meaningful way—my notation for units is justified as follows: d stands for "dualising" object; o stands for "original" (=initial) object; t stands for "terminal" object; and e is simply a generic symbol for neutral elements.]

2.5 Yet one more option: `yom`

By way of contrast, my only criticism of the quantale-theoretic notation is that the symbols $\wedge, \vee$ are too closely associated with posets to be used for arbitrary categories. In particular, since categorical models of linear logic often happen to be order-enriched, it seems sensible to reserve $\wedge, \vee$ to denote the meet and join of parallel arrows, when they exist.

Thus, while I consider `cjm` fine for the syntax of linear logic, I briefly—*i.e.*, prior to developing the notation of option `jme`—considered the following notation for the categorical semantics of linear logic:

`x \tens y`	$\mapsto$	$x \binampersand y$	`\uftens`	$\mapsto$	e
`x \parr y`	$\mapsto$	$x \bindnasrepma y$	`\ufparr`	$\mapsto$	d
`x \with y`	$\mapsto$	$x \sqcap y$	`\ufwith`	$\mapsto$	t
`x \plus y`	$\mapsto$	$x \sqcup y$	`\ufplus`	$\mapsto$	o

and therefore include it in `llconnex` as an interesting relic.

2.6 Abbreviated macros for units: `uns`

My preferred notation for nullaries suggests the following abbreviations:

`\uftens`	$\mapsfrom$	`\e`
`\ufparr`	$\mapsfrom$	`\d`
`\ufwith`	$\mapsfrom$	`\t`
`\ufplus`	$\mapsfrom$	`\o`

—these can be loaded by including option `uns` in addition to one of the options listed above.

Note, however, that `\d`, `\o` and `\t` already have meanings in TeX: by default,

`\d{a}`	produces	ạ
`\o`	produces	ø
`\t{a}{a}`	produces	a͡a

—the option `uns` simply overrides these.

A user who needs one of these symbols should either not use this option, or include something along the lines of

```
\let\udot\d     \let\nordico\o
```

at a point prior to `\usepackage[...,uns]{llconnex}`.

[Strangely, however, `\let\tieover\t` does not seem to have the desired effect.]

2.7 Indexed connectives

It is rare that one needs to deal with infinitary conjunctions and disjunctions in linear logic—although this can occur, for instance in quantale theory.

Nevertheless, it seemed sensible to include "large" versions of each of the connectives described above, as these can also be used for finite indices—*e.g.*, $\bigparr_{j=1}^{n} x_j$.

The macros for these large connectives are `\bigtens`, `\bigparr`, `\bigwith` and `\bigplus`; in each case they produce a version of the ordinary binary symbol with appropriate spacing, except in option `cns` where `\bigwith` compiles to $\prod$, and `\bigplus` compiles to $\sum$.

3 Negations and Implications

In linear logic proper there exists only one notion of linear negation, but in non-commutative linear logic, there exist two. Wishing to be all things to all people, `llconnex` accommodates the possibility of two linear negations, and therefore also of two linear implications.

linear negation (*alias*: perp)	$\mapsto$	`\perp`
reverse linear negation	$\mapsto$	`\prep`
linear implication (*alias*: lollipop)	$\mapsto$	`\loll`
reverse linear implication	$\mapsto$	`\llol`

By default, these compile as follows:

$$\begin{array}{rcl} \texttt{\textbackslash perp\{x\}} & \mapsto & x^{\perp} \\ \texttt{\textbackslash prep\{y\}} & \mapsto & {}^{\perp}y \\ \texttt{x \textbackslash loll z} & \mapsto & x \multimap z \\ \texttt{z \textbackslash llol y} & \mapsto & z \circ\!\!- y \end{array}$$

but options can be used to change these.

Note that `\perp` and `\prep` take an argument; either used alone will produce an error. When referring to linear negation as a functor, it is advisable to pre-define a blank space (*e.g.*, `\def\blank{{(-)}}`), so that one can simply write `\perp\blank`.

3.1 Alternatives for negations

Many category theorists, such as M. Barr and myself, prefer to denote linear negations with asterisks, as is standard in duality theory. We therefore include an option `ast`, which produces the following results:

$$\begin{array}{rcl} \texttt{\textbackslash perp\{x\}} & \mapsto & x^{*} \\ \texttt{\textbackslash prep\{y\}} & \mapsto & {}^{*}y \end{array}$$

Other category theorists, such as A. Joyal and R. Street, apparently object to having superscripts on the left. Option `jns` produces their notation, which is as follows:

$$\begin{array}{rcl} \texttt{\textbackslash perp\{x\}} & \mapsto & x^{*} \\ \texttt{\textbackslash prep\{y\}} & \mapsto & y^{\vee} \end{array}$$

Another convention for avoiding superscripts on the left, is provided by option `mus`:

$$\begin{array}{rcl} \texttt{\textbackslash perp\{a\}} & \mapsto & a^{\sharp} \\ \texttt{\textbackslash prep\{b\}} & \mapsto & b^{\flat} \end{array}$$

—anyone who has played the piano will find this natural.

3.2 Alternatives for implication

The lollipop symbols for linear implication are nearly universal. The only exceptions occur in quantale theory papers, where "residuations" are commonly denoted with ordinary arrows.

Option `arr` produces

`x \loll z`	$\mapsto$	$x \rightarrow z$
`z \llol y`	$\mapsto$	$z \leftarrow y$

and option `Arr` produces

`x \loll z`	$\mapsto$	$x \Rightarrow z$
`z \llol y`	$\mapsto$	$z \Leftarrow y$

4 Exponentials

Options for exponentials are redundant, as all authors seem to agree on how to denote them: ! for the negative one (*alias*: bang, of course, *bien sûr*), and ? for the positive one (*alias*: whimper, why not, *pourquoi pas*).

The macros for these symbols are

bang, of course, *bien sûr*	$\mapsto$	`\bang`, `\duhh`, `\bsur`
whimper, why not, *pourquoi pas*	$\mapsto$	`\whim`, `\ynot`, `\ppas`

5 Other connectives

Future versions of `llconnex` may incorporate macros for the linear co-implications, and for connectives appearing in extensions of linear logic, *e.g.* the non-commutative *seque* operations.

6 Symbols new and old

6.1 Superimposed math symbols

Only four symbols defined in `llconnex` are entirely new: the binary "partial tensor product" symbols, $\ntimes$ and $\utimes$ and their large versions, $\bigntimes$ and $\bigutimes$. These can, at any time, be produced using the macros `\ntimes`, `\utimes`, `\bigntimes` and `\bigutimes`, respectively. Understanding how these symbols are created will allow the user to create many more similar symbols—$\boxtimes$, $\mathsuperimpose{\circ}{\fixcup}$, *&c*—should the need arise.

Firstly, `llconnex` defines a symbol `\fixtimes`, which is simply a smaller version of $\times$ and a symbol `\fixcup` which is a slightly lower version of $\cup$. Next, `llconnex` creates a means of superimposing two math symbols, `\mathsuperimpose`. Finally, `llconnex` defines `\ntimes` and `\utimes` by `\mathsuperimpose`-ing a `\fixtimes` on a `\cap` and a `\fixcup`, respectively;

`\bigntimes` and `\bigutimes` are defined by `\mathsuperimpose`-ing an ordinary `\times` on a `\bigcap` and a `\bigcup`, respectively.

Thus $\mathsuperimpose\fixtimes\sqcap$ can be obtained by `\mathsuperimpose\fixtimes\sqcap` and $\mathsuperimpose\circ\fixcup$ by `\mathsuperimpose\circ\fixcup`. Note that the wider of the two symbols (if they are not equal in width) should be placed as the second argument of `\mathsuperimpose`.

6.2 Ampersand and ampersor

The symbols $\binampersand, \bindnasrepma$ are part of the `stmaryrd` package, which is required for `llconnex`.

There they go under the names `\binampersand` and `\bindnasrepma`, respectively, but `llconnex` defines `\amper` and `\repma`, to mean better-spaced versions of the same.

```
\def\amper{\mathrel{\binampersand}}
\def\repma{\mathrel{\bindnasrepma}}
```

It also defines `\bigamper` and `\bigrepma` for indexed conjunctions and disjunctions.

```
\def\bigamper{\mathop{\binampersand}\limits}
\def\bigrepma{\mathop{\bindnasrepma}\limits}
```

6.3 Others

The symbol $\nplus$ also exists in `stmaryrd.sty`, where it goes under the name `\nplus`; and $\uplus$ is, of course, standard under the name `\uplus`.

The definitions of `\loll` and `\llol` were stolen from R.A.G. Seely's list of macros, `ragsmac.tex`; `llconnex` also defines a number of similar symbols:

`\sqloll`	$\mapsto$	$\sqloll$	`\sqllol`	$\mapsto$	$\sqllol$
`\varloll`	$\mapsto$	$\varloll$	`\varllol`	$\mapsto$	$\varllol$
`\doubloll`	$\mapsto$	$\doubloll$	`\doubllol`	$\mapsto$	$\doubllol$.

7 Bugs, tips, *&c.*

The definition of `\fixtimes` uses hard measures which are optimised for 12pt fonts; these can, however, be redefined. For example, if using 11pt font, I recommend `\def\quasipt{0.75pt}` and, if using 10pt font, `\def\quasipt{0.67pt}` to produce better results for $\mathsuperimpose\fixtimes\cap$ and $\mathsuperimpose\fixtimes\cup$. These commands should appear immediately after, not before, `\usepackage[...]{llconnex}`.

Moreover, the scriptscript-size version of `\fixtimes` is simply a dot, which is less than ideal. Thus, for example, `{\scriptscriptstyle x \ntimes y}` produces only ${\scriptscriptstyle x \ntimes y}$. Hopefully, this should not cause many problems. A useful tip, when using Xy-pic, is to include the command `\let\labelstyle\textstyle` to increase the size of arrow names.

If you appreciate `llconnex`, then keep an eye out for `mnats.sty` which will attempt to do for those natural transformations which commonly arise in monoidal category theory what `llconnex` does for functors.

References

[1] Michael Barr. *$*$-autonomous categories*, volume 752 of *Lecture Notes in Mathematics*. Springer, Berlin, 1979. With an appendix by Po Hsiang Chu.

[2] J. R. B. Cockett and R. A. G. Seely. Weakly distributive categories. *J. Pure Appl. Algebra*, 114(2):133–173, 1997.

[3] Jean-Yves Girard. Linear logic. *Theoret. Comput. Sci.*, 50(1):101, 1987.

[4] Christopher J. Mulvey. &. *Rend. Circ. Mat. Palermo (2) Suppl.*, (12):99–104, 1986. Second topology conference (Taormina, 1984).

[5] Kimmo I. Rosenthal. *Quantales and their applications*, volume 234 of *Pitman Research Notes in Mathematics Series*. Longman Scientific & Technical, Harlow, 1990.