frege.sty
A LaTeX 2ε Package for Typesetting Begriffsschrift

Quirin Pamp
Quirin.Pamp.2009@my.bristol.ac.uk

August 4, 2012

1 Background

1.1 Motivation for this Package

I recently decided to read Frege's *Begriffsschrift* from 1879 and found that the only copy I could find online was a rather poor quality scan of the original. Since the copyright on german publications expires some 70 or so years after the death of their author, I had the bright idea to combine reading the paper with the making of an electronic copy. This required the typesetting of begriffsschrift. A quick search on the internet assured me that there was a LaTeX package for just this purpose, and off I went. However I quickly noticed that it would be very difficult to achieve a typesetting I deemed sufficiently close to the original using only the package *begriff* by Josh Parsons. Despite the fact that I have never written a LaTeX package before, a quick look at the source file (only some 300 or so lines with plenty of comments) along with some head scratching convinced me to embark on this further project.

With the help of the begriff package and the good people on StackExchange, I eventually produced a package that was able to do everything that the original begriff package can, albeit with a few changes I consider an improvement. Further versions largely reflect additional features I added as I continue to type up Frege's original paper, as well as bug fixes and input

by members of the LaTeX community. Once I am done I will add the tex file for Fege's paper as the definitive example for the usage of this package.

1.2 The *begriff* Package

This package is based on begriff.sty released under the GNU General Public License. Copyright (C) 2003 by Josh Parsons (josh@coombs.anu.edu.au) with changes made in October 2004 by Richard Heck (heck@fas.harvard.edu) and minor changes by Josh Parsons to fix a problem with linespacing made in May 2005.

While I could not have done so without the aforementioned work, I have reworked the package from the ground up, to the point where some of the underlying approaches have changed. On the downside this means there is no simple way of converting anything typeset using begriff.sty to use this package instead. I felt this was necessary to achieve an end result I was happy with.

1.3 The *bguq* Package

Since version 1.3 I added the option to use the bguq character for all quantifiers. This character is provided by the bguq package by J.J. Green.

2 Version History

2.1 Changes as Compared to begriff.sty

- correct (closer to the original typesetting) relative lengths of the content stroke with respect to other strokes attached to it;
- content strokes point at the middle of the following symbols, rather than the bottom;
- greater width for the assertion stroke as compared to the content stroke;
- a more intuitive structure for the conditional (arguments are now given in the same order as they appear on the left of a completed formula);
- the command for the conditional with empty arguments now results in a vertical line (conditional stroke) on it's own the other strokes are added as the arguments;

- the linewidth is properly accounted for so that things remain properly centered when scaled;

2.2 Changes in Version 1.1

- added an optional scale factor to all basic strokes;
- simplified the code for Fbracket in terms of that for Fbox;
- rearranged the code in the style file in a more logical way;

2.3 Changes in Version 1.2

- added a new command "Fargument" for typesetting arguments;
- added a new command "Fstrut" to be used in conjunction with Fargument;
- changed Fbaselength to be equal to the full length of a basic stroke (20pt);
- fixed a bug where the scale factor introduced in version 1.1 does not reset after all uses;

2.4 Changes in Version 1.3

- made the "():" used in the Fargument command introduced in version 1.2 user defined so as to make the option properly optional;
- added a strut to Fargument so as to produce visually correct centering;
- added optional shorthands for all commands for a better flow of usage;
- added the option "bguq" to the package which uses the bguq font by J.J. Green for all quantifiers;

2.5 Changes and Features yet to come

- a way to display nested arguments;
- the fregean conjunction (it exists);
- a vertical shorthand stroke such as used by Frege for typesetting arguments in his original paper;
- scaling symbols automatically with changes in font size;
- scaling of the bguq character with changes in Flinewidth;

3 Features and Usage

3.1 Commands

3.1.1 Basic Commands

The following is a list of the basic commands provided by this package along with accompanying output and the optional shorthand for the command.

Command:	Output:	Shorthand:
\Fcontent[1]		\F[1]
\Fncontent[1]		\Fn[1]
\Fnncontent[1]		\Fnn[1]
\Facontent[1]		\Fa[1]
\Fancontent[1]		\Fan[1]
\Fanncontent[1]		\Fann[1]
\Fquant[1]{a}		\Fq[1]
\Fnquant[1]{a}		\Fnq[1]{a}
\Fnnquant[1]{a}		\Fnnq[1]{a}
\Fquantn[1]{a}		\Fqn[1]{a}
\Fquantnn[1]{a}		\Fqnn[1]{a}
\Fnquantn[1]{a}		\Fnqn[1]{a}
\Fnquantnn[1]{a}		\Fnqnn[1]{a}
\Fnnquantn[1]{a}		\Fnnqn[1]{a}
\Fnnquantnn[1]{a}		\Fnnqnn[1]{a}
\Faquant[1]{a}		\Faq[1]{a}
\Fanquant[1]{a}		\Fanq[1]{a}
\Fannquant[1]{a}		\Fannq[1]{a}
\Faquantn[1]{a}		\Faqn[1]{a}
\Faquantnn[1]{a}		\Faqnn[1]{a}
\Fanquantn[1]{a}		\Fanqn[1]{a}
\Fanquantnn[1]{a}		\Fanqnn[1]{a}
\Fannquantn[1]{a}		\Fannqn[1]{a}
\Fannquantnn[1]{a}		\Fannqnn[1]{a}

This may seem like a daunting list, but there is an exceedingly simple way to think about it. In a sense there are only two commands \Fcontent[1] and \Fquant[1]{}. These two commands can be augmented with a combination of as and ns so as to add assertion and negation strokes respectively. Any stroke that is asserted (has a fat vertical line at the start) starts with '\Fa'. This may be followed by either one or two or no 'n' to add one or two or no negation strokes (the small vertical lines below the content stroke). Next comes the name of the main command, either 'quant' or 'content'. Finally the quantifiers may be followed by either one or two 'n' to add one or two negation strokes to the content stroke after the quantifier's depression.

Consider also that many of these commands are only really present for completeness sake. It is difficult to imagine a situation where a twice negated quantifier with twice negated content would ever be needed.

Since version 1.1 all basic strokes also have an optional scaling factor. A command followed by [.5], for example would produce a stroke exactly half the default length while [2] produces a stroke twice the default length. (The default length is given by \Fbaselength which is set to 20pt. Scaling allows for greater control in the total length of formula as well as for a shorter syntax. (We can replace expressions like \Facontent \Fcontent with \Facontent[2].) Care must be taken not to set a length that is shorter than what is needed to fit all the parts of some basic stroke. This will lead to negative lengths and hence unpredictable output.

All quantifiers also have a mandatory argument that specifies the variable associated with the quantifier. (Mandatory arguments are contained in a set of curly brackets { and }). This argument should be a single small letter and will be typeset above the semi circular depression in the assertion stroke in mathfrak font which is provided by tha amssymb package. This font can be used in maths mode by using the command '\mathfrak{}'. Note that all the commands provided by this package may be used in both math and text mode. (Though math mode usually results in better formatting.)

Finally one may combine the above commands in arbitrary combinations which will result in gapless longer strokes. (Eg.: $\mathfrak{a}\ \mathfrak{b}\ A$) which may be roughly translated into english as "for all $\mathfrak{a}$ there exists a $\mathfrak{b}$ such that A". (The commands I used for this expression are $\Faquant{a}\Fnquantn{b}A$).

3.1.2 Conditional

The conditional is the most important command in this package since it gives Frege's Begriffsschrift it's two dimensional structure.
The syntax for the Fconditional command is as follows:

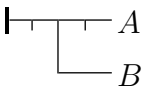
\Fconditional[<option>] {<consequent>}{<antecedent>}

The shorthand version (since version 1.3) is given by "\Fcdt".

The arguments may in principle be anything, but you will only get a begriffschrift formula if the arguments are themselves given by appropriate commands from the list of basic commands given earlier. As an example, an asserted conjunction between A and B would be given as follows:

$\Fconditional[\Fancontent]{\Fncontent A}{\Fcontent B}$

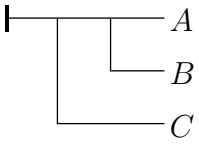
and produce the following output:



In addition Fconditional may be nested as it's own argument to arbitrary depth. Nesting in the option is not recommended.
A conditional with nested consequent may be given as follows:

$\Fconditional[\Facontent]{\Fcontent\Fconditional{\Fcontent A}{\Fcontent B}}{\Fcontent\Fcontent C}$

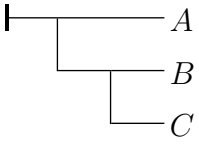
and produces the following output:



A conditional with nested antecedent may be given as follows:

$\Fconditional[\Facontent]{\Fcontent \Fcontent A}{\Fcontent\Fconditional {\Fcontent B}{\Fcontent C}}$

and produces the following output:



Each section of a content stroke may thus be replaced with any of the strokes given by the list of basic commands. Note that it is up to the user to place the

appropriate number of strokes in each argument to ensure that the content strokes all line up on the right hand side.

3.1.3 Brackets and Boxes

There are two more commands to be considered:

\Fbox{<complex expression>}

The shorthand version (since version 1.3) is given by \Fb{}

\Fbracket{<complex expression>}

The shorthand version (since version 1.3) is given by \Fbb{}

Both Fbox and Fbracket take what I have called a 'complex expression' for their argument. A 'complex expression' is any formula in begriffsschrift that has at least one conditional in it. It is generally a good idea to put all complex expressions into either a Fbox or a Fbracket. It is never necessary to place a complex expressions into both an Fbox and an Fbracket since an Fbox simply is a Fbracket without the actual brackets. Fbracket exists only for convenience with the same effect being achieved by \left(\Fbox{} \right).

The reason why the Fbox is a good idea, is that the baseline is very near the top of a complex expression of Begriffsschrift, which can make for some odd formatting effects. In addition to placing the baseline at the middle of a complex expression an Fbox ensures the expression is treated by LaTeX as a single object and given enough space.

Finally a complex expression may not format correctly in some environments (like the align* environment for example) unless it is placed in an Fbox. In short, always use an Fbox (or Fbracket).

3.1.4 Arguments and Struts

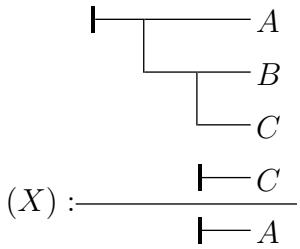
Since version 1.2 two commands have been added that allow for the typesetting of arguments. The syntax for the argument command is as follows:

\Fargument[<premise 0>]{<premise 1>}{<premise 2>}{<conclusion >}

The shorthand version (since version 1.3) is given by \Farg

In the following esample the optional argument for premise 0 (an absent premise takes the value '(X) :' the premises are the two formulas above the therfore line and the conclusion is the formula below the therefore line. The Begriffschrift expressions in the arguments of the Fargument command do not need to be placed in an Fbox, since the Fargument command works by

boxing it's arguments allready.



where $X = \vdash B$; (this is typeset seperately from the Fargument command);

The three begriffsschrift formulas above are in fact aligned leftbound. To make them appear rightbound no matter what the relative lengths of A, B, and C, the command "Fstrut" has been used in front of $\vdash C$ and $\vdash A$. The command "\Fstrut[1]" works exactly like an invisible content stroke, that is it inserts space of length Fbaselength. Like all basic strokes it can be scaled via an optional scale factor. Since version 1.3 it may be called by the optional shorthand "\Fs"

3.2 Lengths

In theory all the dimensions in this package can be changed with the command \setlength{<name of length>}{<new value>}, though I have not done a great deal of testing and recommend sticking to the default values. The following then is a table of all lengths:

name	default value	description
\Fbaselength	20pt	the length of the basic strokes
\Flinewidth	0.5pt	the line width
\Fspace	2pt	seperation between lines and text/formula
\Fassertwidth	3\Flinewidth	width of assert stroke
\Fraiseheight	1ex-\Flinewidth	height of content lines above baseline
\Fnegsep	3\Flinewidth	seperation between a double negation
\Fnegshort	2\Flinewidth	space between negation stroke and baseline
\Fquantwidth	6pt	width of the semi-circular quantifier depression

the height of the conditional stroke is determined by the size of the contents of the conditionals argument, as well as the baselineskip of the surrounding text. It cannot be changed manually.

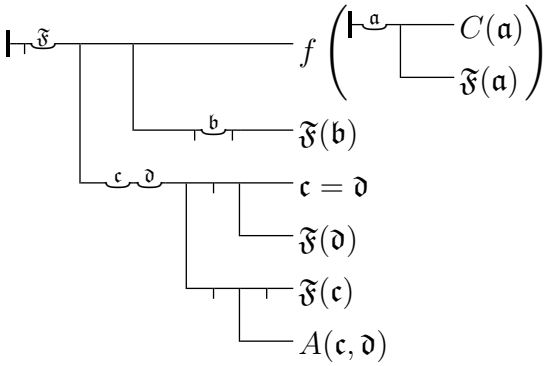
3.3 The *bguq* Option

Since version 1.3 this package may be called with the option "bguq" as follows:
\usepackage[bguq]{frege}
If the option is enabled all quantifiers will be typeset using the bguq font provided by the bguq package by J.J. Green. (This document is typeset using the option). This means that the bguq package must be installed if the option is enabled. (It can be found on ctan).

Warning: At present the bguq character scales with font size while the rest of the symbols provided by this package do not. Also the bguq character does not respond to a change in Flinewidth.

3.4 Final Example

The Geach-Kaplan sentance (with thanks to Marcus Rossberg):



And that is all.
For comments, suggestions, identified errors, email me at
<Q.Pamp.2009@my.bristol.ac.uk>.