# frege.sty
# A LaTeX 2ε Package for Typesetting Begriffsschrift

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

July 14, 2012

## 1 Background

### 1.1 begriff.sty

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 a end result I was happy with.

### 1.2 GNU General Public License

This package and all accompanying documentation is released under the GNU General Public License. It is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published

by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111, USA. (or just search for it online... Seriously.)

## 1.3 Changes as Compared to begriff.sty

- correct relative lengths of the content stroke with respect to other strokes attached to it;
- content strokes point at the middle of the followint 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;

## 1.4 Changes and Features yet to come

- incorporation of other frege related material such as fge.sty so as to put all frege related material in a single package;
- a therefore stroke for typesetting arguments;
- a vertical shorthand stroke such as used by Frege for typesetting arguments in "Begriffsschrift";
- reducing the length of the content stroke between the depressions of two adjacent quantifiers;

# 2 Features and Usage

## 2.1 Commands

### 2.1.1 Basic Commands

The following is a list of the basic commands provided by this package along with accompanying output. The conditional is treated sepperately.

\Fcontent 

\Fncontent 

\Fnncontent 

\Facontent 

\Fancontent 

\Fanncontent 

\Fquant{a} 

\Fnquant{a} 

\Fnnquant{a} 

\Fquantn{a} 

\Fquantnn{a} 

\Fnquantn{a} 

\Fnquantnn{a} 

\Fnnquantn{a} 

\Fnnquantnn{a} 

\Faquant{a} 

\Fanquant{a} 

\Fannquant{a} 

\Faquantn{a} 

\Faquantnn{a} 

\Fanquantn{a} 

\Fanquantnn{a} 

\Fannquantn{a} 

\Fannquantnn{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 and \Fquant{}. 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 'n' to add one or two negation strokes (the small vertical lines below the assertion 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.

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.: $\vdash$ $\mathfrak{a}$ $\neg$ $\mathfrak{b}$ $\neg$ $A$) 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\$).

### 2.1.2 Conditional

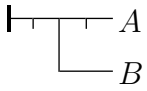
The conditional is the most important command in this package since it gives Frege's Begriffsschrift it's two dimensional structure.
The syntax for the Fconditional command is as follows:

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

The 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}\$

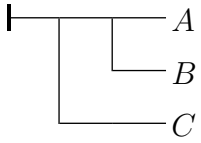
and produce the following output:




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:

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

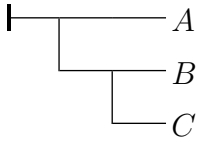
and produces the following output:




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.

### 2.1.3 Brackets and Boxes

There are two more commands to be considered:

\Fbracket{<complex expression>}

\Fbox{<complex expression>}

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 middel 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).

## 2.2 Lengths

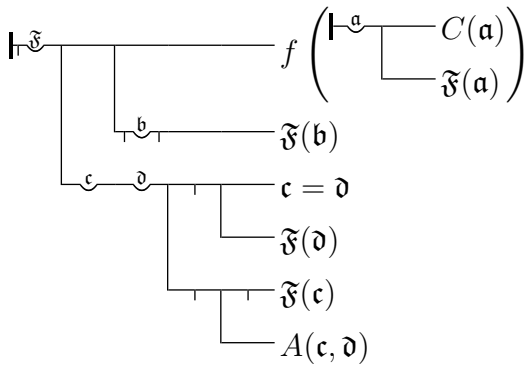
In theory all the dimensions in this package can be changed with the command \setlength{<name of length>}{<new value>}, though I have not done a great deal of testing and recommend sticking to the default values. The following then is a table of all lengths:

| name | default value | description |
|---|---|---|
| \Fbaselength | 10pt | half the length of a basic stroke |
| \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.

## 2.3 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>.