Newmath and Unicode

Johannes Küster*

Abstract

The "Newmath" project aims at defining and implementing new standard encodings for math fonts, and at the development of accompanying tools and packages. Switching math fonts should be made as easy as switching text fonts. The project stopped in 1998, as efforts were concentrated on the definition of Unicode codepoints for mathematics.

This article outlines my ideas for further development of Newmath. It deals mainly with the "encodings and fonts" part of the project. Originally the project aimed only at extending and reorganizing the encodings of existing math fonts, but its objectives should be widened now to Unicode math – to make all those mathematical characters accessible and usable in TEX-based systems. The last section gives an outlook on Latin Modern Math fonts development.

Introduction

About 12 years ago, the Math Font Group[1] (MFG) started a project to define new standard encodings for TEX math fonts, together with the development of fonts implementing this new standard and of accompanying tools and packages. The new encodings should bring an extension to 256 codepoints per font. They should become the standard for TEX math fonts, ideally making it just as easy to switch between different math fonts as it has been achieved for text fonts. This whole project is called "Newmath", short for New Math Font Setup ("newmath.sty" is the name of the principal LATEX package implementing the new math font setup).

The development of Newmath stopped about 6 years ago, as it was decided (at the EuroTEX conference in St. Malo in 1998) to concentrate efforts first on "Math into Unicode", i.e. to identify all mathematical symbols in (reasonable) use and to get these symbols encoded in the Unicode standard. This goal has been achieved for quite a while now, mainly with Unicode 3.2 in 2002, but work on the Newmath encodings has not been resumed since (mainly because the people originally involved quit for other projects in the meantime).

Is further development of Newmath still interesting at all, despite Unicode and OpenType fonts? I think it is, for the reasons discussed below. But its initial objectives should be widened: to make all Unicode math characters accessible in TEX in a standard way, but also to make math fonts easier to design or to adapt, and to make them more usable for other typesetting systems.

Current State of Newmath

Partial implementations of the new Math Font Encodings, information about the development of Newmath, links to articles, conference presentations, mailing list archives etc. can be found at the MFG homepage [1]. The development of Newmath stopped in 1998 with version 0.59a. The implementation was mainly done by Matthias Clasen with help from Ulrik Vieth.

* Author's address: typoma; Karl-Stieler-Str. 4, D-83607 Holzkirchen, Germany; info@typoma.com; http://www.typoma.com

1 The Math Font Group is a joint venture of the LATEX3 project and the TEX Users Group Technical Working Group on Extended Math Font Encoding. For more information see the Math Font Group's homepage [1].

Currently six encodings are defined:

- *Math Core* (MC) contains Math Italic, Greek Upright and Italic, basic delimiters and other "alphabetic" characters (i.e. most of the characters which are really dependent on the font design and/or which are most likely to pre-exist in a text font)
- *Math Symbol Principal* (MSP)[2] contains a Calligraphic (or Formal Script) alphabet, the most important mathematical symbols, and basic accents
- *Math Symbol 1* (MS1 or MSA) contains a Blackboard Bold (or "Doublestroke") alphabet and additional mathematical symbols
- *Math Symbol 2* (MS2 or MSB) contains a Fraktur (or Blackletter) alphabet, some additional delimiters and accents, and an "Arrow Kit" (consisting of left and right arrow endings and repeatable middle parts (with negated and gapped versions), by which a great variety of different arrows at any desired length can be composed)
- *Math Extension Principal* (MXP)[2] contains text and display versions of big operators and integrals, wider version of basic accents (hat and tilde), the larger and extensible versions of the basic delimiters, larger root symbols, over- and underbrace parts, parts for extensible vertical arrows and bars
- *Math Extension 1* (MX1 or MXA) contains additional big operators and integrals, larger and extensible versions of delimiters, and wider accents (vector, bar, tie, etc.).

Each of the *Symbol* encodings contains a complete alphabet (A–Z, a–z, digits 0–9, dotless i and j) of a specific design for use in mathematical typesetting.

Compared with TEX's original "Math Extension" encoding, the new extension font encodings offer a much wider range of wide accents (9 sizes of each accent) and large delimiters (7 sizes instead of 4 for most delimiters, 14 for parentheses and non-extensible delimiters like angle brackets, plus extensible parts as necessary).

Unicode Math

Since version 3.2, Unicode assigns almost 2.000 codepoints for mathematical characters. Due to the way in which Unicode evolved, and as new versions should be backward compatible, these codepoints are scattered over many Unicode blocks (mainly over 15 blocks in fact, of which 7 blocks are devoted exclusively to math characters; another 15 blocks each contain a few characters for occasional use in mathematics).

Additional information about Unicode math is given in the Unicode Technical Report #25 [2] and in "MathClass6.txt" [3], a file which classifies the Unicode math characters according to their usage and provides "a mapping to standard entity sets commonly used for SGML and MathML documents". The classification is comparable to TEX's mathematical symbol classes, with the additional classes "diacritic" (which is not handled as a class by TEX, but by `\mathaccent`) and "fence" (an unpaired delimiter or a delimiter-like separator; normally treated as `\mathrel` in TEX).

Glyph Variants in Unicode. Some mathematical symbols did not get a codepoint of their own, instead they can be accessed as a combination of two Unicode codes (examples are shown below). This is the case for the negated version of many relators, for variants of negated relators (with a vertical negation slash instead of a slanted one), and for some symbols which are considered mere stylistic or typographic variants of another symbol.

These variants are shown in three tables in [2]: Table 2.5 there shows those relators with encoded negated form for which a variant with vertical stroke overlay can be realized by composition of base character and overlay; Table 2.6 shows those relators for which the negated form can only be realized by composition (i.e. the negated form is not encoded itself); and Table 2.7 shows all the currently defined glyph variants, which can be realized as a combination with "Variant Selector 1" U+FE00.

2 Originally, the Math Symbol and Math Extension "Principal" encodings were named "Primary"; later this was changed to "Privilege". As I think neither really conveys the intended meaning, I decided to change the name to "Principal".

∉	U+2209	∉	U+2208, 20D2	NOT AN ELEMENT OF
≯	U+2267, 0338	≯	U+2267, 20D2	NEGATED GREATER-THAN OVER EQUAL TO
≨	U+2268	≨	U+2268, FE00	LESS-THAN BUT NOT EQUAL TO

Examples of Unicode variants and combinations: with U+20D2 "vertical line overlay", with U+0338 "combining long solidus overlay" (with and without encoded negated version), and with U+FE00 "Variant Selector 1"

Future Additions to Unicode Math. Of course Unicode math can be and will be extended in the future. At the time of writing, the current version of Unicode is 4.1.0 (released March 2005). A few math characters were added in this version. Furthermore, the Unicode Pipeline Table [4] shows some mathematical characters which are currently under consideration for future inclusion in Unicode. Also newly discovered and newly invented symbols will be standardized in future version eventually, when they are "used by a considerable number of people." Such possible extension have to be taken into account when designing new math font encodings for TEX.

Reasons for Newmath

To see more clearly how Unicode math could be made usable for TEX, and why Newmath could still be very helpful, let's see what Unicode does offer and what it does not.

Unicode offers a very large set of mathematical characters, with a standardized code referring to each character. But the backward-compatibility leads to a quite unordered way in which characters are presented within Unicode. This is not a problem for a computer program (e.g. for any automatic conversion program, workflow processes and the like), but it makes it difficult for users to search for a specific character, or for font designers to get an overview over all those math characters.

Unicode does not offer any sorting or ranking of mathematical symbols, nor much information about the importance, meaning or usage of most characters. Also Unicode encodes only base characters, thus leaving all typographic variants aside which are needed in proper mathematical composition. Now theoretically all those glyphs could be defined in one large OpenType font, by assigning glyphs in the Private Use Areas (PUA) of Unicode, and/or by defining those glyphs as alternate forms of their base glyph via OpenType features. Unfortunately, it is not very likely that a standard way of PUA usage will evolve for math fonts. Also, the currently defined OpenType features are hardly suitable or sufficient for math fonts.

So I see many reasons why Newmath is still interesting and could be useful, despite Unicode and OpenType, and despite any successor of TeX which will be Unicode and OpenType capable:

- Newmath offers a standard interface for TEX (LATEX, ConTEXt).
  Currently almost each set of math fonts comes with its own encodings, which makes font switching very cumbersome.
- Newmath will offer all the typographical variants needed.
  This comprises most of the characters in extension fonts: larger and extensible delimiters, arrows and root symbols; text and display versions of big operators and integrals; wide accents. (This could be done in an OpenType font as well, of course.)
- Newmath will order, sort and rank mathematical characters.
  This will give a much better overview than it is possible in Unicode, making it comparatively easy to find a specific character, to judge its importance, etc.
- Newmath could serve as a guideline to font designers.
  Within Unicode, it is very hard for a font designer to identify the characters needed for mathematics, and to seperate indispensable math characters from less important ones. In fact, most font designers will be abhorred by the prospect of designing 2000 additional characters, of which many will be seldom used.

Here Newmath could offer a clearly arranged and well-ordered set. For example is the math asterisk ∗ (U+204E LOW ASTERISK) easily overlooked, as most fonts already contain an asterisk * (U+002A ASTERISK), but normally the latter is not suitable for math, which uses a larger version, six-pointed and vertically centered at the mathematical axis.

– Newmath could be used to classify math fonts.
Currently it is not easy to judge the usability of a specific math font, and to gain a quick overview of its glyph complement set. This could be made much easier by classifying the font according to the set of its supported Newmath encodings. For example, it should be quite easy then to see that a font has all the Unicode glyphs needed in Mathematical Logic.

Further Development of Newmath

For further development of Newmath, I see the following areas: *Encodings; macros; fonts; packages and tools; integration and interaction with other packages; additions and enhancements to TEX's mathematical typesetting engine.*

I am mainly concerned with encodings, macros and fonts here, and my ideas for these areas are detailed in the sections below.

The development of "packages and tools" will, for a good part, go hand in hand with the development of macros (for LaTeX, ConTeXt and plain TeX). By "integration and interaction with other packages" I mean that Newmath should work with other math packages (e.g. `amsmath` or `nath` in LaTeX), but also that Newmath could borrow and integrate from other packages (e.g. macros in widespread use could be standardized). For possible "additions and enhancements to TeX's mathematical typesetting engine", see Ulrik Vieth's article [5]. Here I'm only dealing with these aspects in the way they influence possible encodings.

Of course I won't and can't do all the necessary work alone, so anyone who wishes to help and to contribute is invited to join the project. Also all steps in the development will be discussed on the Math Font Group's "math-font-discuss" mailing list (see [1] for information about the mailing list and how to join it).

Development of the Encodings

General Considerations. We have to take TeX's restrictions into account: only 16 families of math fonts are allowed in one formula (practically, this means in one document in most cases). Therefore the additional encodings should be designed in a way that minimizes the loading of additional fonts.

In a TFM file, only 15 different non-zero heights and 15 non-zero depths are allowed. While one could cope with this for symbol fonts in most cases, it is a really troublesome restriction when it comes to extension fonts. This leads to the strange vertical placement of most glyphs in extension fonts, which hinders their usability outside of TeX. But even within TeX, it could become impossible to cope with for some fonts which differ in design from some of Computer Modern's assumptions.

These restrictions should be overcome by any successor of TeX, maybe best with a new "math font metrics" format, but for the time being, the encodings should deal with them as good as possible.

But the encodings and macro packages should not be tied too closely to TeX and the current situation; they have to be flexible enough to be extendable – to other typesetting traditions like those of traditional Russian mathematical typography, and to font sets which bring their own extensions and special macros, like the MathTime Pro fonts.

The Existing Encodings. I consider *Math Core* as fairly stable. Maybe about 5 characters could be moved to another encoding.[3] This would allow to include a few Roman characters like e, i, and maybe D (MathCore

3 For example, only italic variants of *ƒ*, *ə*, and *ɪ* are encoded here; moving these to an additional font-dependent encoding would allow to encode their upright variants there as well.

already contains "d", and these single Roman letters are quite common in mathematical typography; inclusion in MathCore would allow kerning with Math Italic letters); but this may sacrifice compatibility with the old math font setup.

Both *Math Symbol Principal* and *Math Symbol One* are stable as well, maybe with one or two questionable characters in each one, and with about 40 yet unassigned codepoints in MS 1. The additional Unicode symbols supply some obvious candidates for inclusion here.

For *Math Symbol Two*, I think that the Fraktur digits are a misunderstanding (the "Fraktur digits" currently included here stem from the Euler fonts, but apparently these are intended as Euler Roman old-style digits; while proper Fraktur digits – clearly visually seperate from Roman digits – just do not exist). So old-style (tabular[4]) digits should be put here, at least for math fonts which bring their own digits, like Euler. But in general, putting old-style digits here would make this encoding dependent on the text font used. A better way could be to put old-style digits in an additional, font-dependent encoding (alongside with additional letters and letterlike symbols which don't find a place in MathCore).

The Arrow Kit could be moved to an encoding of its own, as many more arrow pieces could be added then (Unicode features many additional arrows; some of these are candidates for extensible arrows).

The case is different for the *extension fonts:* maybe the whole encodings should be overthrown, maybe we should sacrifice compatibility with older documents here in favour of a clearer layout. By putting e.g. root, accents, and over- and underbrace into one encoding, and putting delimiters and big operators into a second one, most glyphs could be brought to their natural position, which would ease the design and greatly improve the general usability of such fonts.

New Additional Encodings. For the remaining Unicode math characters (i.e. for the characters not yet encoded in Newmath), we have to design new, additional encodings. First, let's see how many additional characters there are, and how many additional forms (like larger delimiters) we need. The following table gives a rough number for the additional characters in each class, with the number of additional codepoints needed in TEX:

Arrows:	250	+ arrow kit pieces:	100	
Binary Operators:	130			
Geometric Symbols:	100			
Miscellaneous Symbols:	30			
Ordinary Symbols:	90			
Punctuation:	15			
Relators:	230	+ negated variants:	100	
Z Notation:	10			
Accents and Overlays:	30	+ in extension fonts:		90
Big Operators:	0	+ in extension fonts:		40
Delimiters:	45	+ in extension fonts:		450
Integrals:	25	+ in extension fonts:		50
Total number of glyphs:	955	+ in symbol fonts:	200	
		+ in extension fonts:		630

This would mean 4 or 5 additional symbol font encodings (possibly including 1 or 2 arrow kit encodings), and 3 or 4 additional extension font encodings.

To minimize the loading of additional fonts, and to offer clearly arranged font layouts (both to users and to font designers), we should sort and group the Unicode characters, according to importance, meaning, and area of use (within mathematics). For example, all symbols specific to one field of mathematics should be kept together in one encoding (e.g. logic, geometry, or z-notation symbols).

4 Many OpenType fonts come with (at least) four sets of digits: lining and old-style, each as proportional and as tabular. For math, tabular digits are used, where all the digits have the same width.

Most documents will only need a limited set of mathematical symbols, and well-designed encodings should make it possible to keep within TEX's restriction to 16 math font families in most cases – without the need for mid-document changes of math encodings.

Unfortunately Unicode does not provide much information neither about the use of a specific character nor about its field of use, so many characters need some research before one could group them properly into an encoding.

Macros and Packages

Along with the new encodings, we do need standard macro names to access those glyphs. Again, any information about the meaning of a character is very helpful here, as then the macro could be named accordingly. For some characters, this is a rather straightforward task.

Ideally, these macro names should be the same in all TEX based systems (especially in plain TEX, LATEX and ConTEXt). Of course one has to develop a "Newmath" package (or file bundle) for each system, but these macro definitions will form the core of each such package and will essentially remain the same.

For LATEX, the current version of Newmath already supplies the necessary files, so we just have to extend these accordingly. When Newmath development ceased in 1998, ConTEXt was not very widespread yet, but now Newmath should of course support it (and vice versa).

TEX's existing math macro names could be broadly categorized as

- descriptive (describing the shape, e.g. `\uparrow`)
- semantic (describing the meaning, e.g. `\sum`, `\times`)
- mixed (partly semantic, partly descriptive, e.g. `\otimes`).

Obviously Knuth employed the following scheme: any symbol with one fixed meaning gets a macro name according to its semantics (thus, `\sum` and not `\bigsigmaup` or whatever). Any symbol without fixed or with more-than-one meaning gets a macro name describing the shape. And the mixed names come in for symbols where the base symbol or a component of the symbol has a semantic name already, but where the meaning of the combined symbol is not clear or not fixed. Of course the new additional macro names should follow this scheme, using a semantic name whenever possible.

In addition, Newmath could be extended to gather macros in widespread use, which could be standardized by including them in Newmath packages. Examples of such macros are `\abs{...}` and `\norm{...}`. Supplying such a standardized set of (alternative) semantic macros could be very helpful to many users. In fact, by using well-chosen semantic macros, a TEX source sometimes can be more readable than its pretty-printed output – and of course it greatly helps in conversion e.g. to Content MathML or OpenMath.

Latin Modern Math Fonts

Encodings, macro packages, and tools are only useful together with fonts. Freely available math fonts could be extended and reencoded, and commercial math fonts could be mapped to the new encodings via virtual fonts (but in most cases they will lack many of the additional glyphs). A good part of this work (on Computer Modern extension and on virtual fonts for other math fonts) has been done already in the last version of Newmath, of course based on the math fonts available at that time. For the future development, it doesn't seem to be a very useful approach to extend the Metafont sources of Computer Modern, as most users would want PostScript Type 1 or OpenType fonts. Instead, I think of extending the Latin Modern fonts.

The Latin Modern math fonts will be a set of freely available math fonts, for use with Latin Modern text fonts. I will design and develop these fonts (with the help of anybody who volunteers to work on these fonts). However I did not start to work on these fonts yet, so I can only give an account of my ideas and intentions here.

I will do the development in MetaType1, so the resulting fonts will be in Type 1 format and could be wrapped as (CFF flavoured) OpenType, too. In general, the design will follow Computer Modern math fonts, with 6 versions of each glyph: 2 weights (regular and boldface) times 3 optical sizes (5 / 7 / 10pt, or rather "Tiny", "Caption", "Regular", as the fonts will be freely scalable of course). But the number of glyphs will be considerably extended, to comprise the complete set of Unicode math characters together with all characters defined in Newmath.

Neither the design nor the metrics of the fonts will be completely compatible with Computer Modern: the design should be more "of a piece" than with Computer Modern and its various extensions (just one example of such a mis-match: the 3 Hebrew letters Beth, Gimel, Daleth from the AMS fonts do not match the design of CM's Aleph, they rather match the Euler fonts' Aleph); and metrics will be changed as needed (e.g. many new kerning pairs will be possibly due to the extended encodings).

By default, these fonts will be encoded in the Newmath standard, thus offering a freely available implementation of the standard. But by the MetaType1 approach the fonts will be independent of any de-facto encoding – so one could rather easy adapt them to further Newmath development, to different encodings, or even to the requirements of other typesetting applications.

References

[1] Math font group project homepage. `http://www.tug.org/twg/mfg/`

[2] Barbara Beeton, Asmus Freytag, and Murray Sargent III. *Unicode technical report #25: Unicode support for mathematics*. `http://www.unicode.org/reports/tr25/`

[3] *MathClass-6.txt – Classification of math characters by usage.* `http://www.unicode.org/reports/tr25/MathClass-6.txt`

[4] *Unicode proposed new characters: The pipeline table.* `http://www.unicode.org/alloc/Pipeline.html`

[5] Ulrik Vieth. Math typesetting in TEX: The good, the bad, the ugly. In: *EuroTEX 2001* (proceedings of the 12th European TEX conference, Kerkrade, the Netherlands), pages 207–216, 2001.