Replacing `cmex`? This is not a finished document

Justin Ziegler
ziegler@educ.emse.fr
ziegler@goofy.zdv.uni-mainz.de

Started: August the 9th
Printed: May 5, 1997
Filename: `cmexfont.tex` with LaTeX and NFSS2

Abstract

While I was working on the new math encoding, I realised that the fact that `cmex` is only loaded in one size, and not in three like the other math fonts, was going to create a few problems. In this paper my concerns are the following:

- What is in `cmex`?
- Which special mechanisms does TeX use to access glyphs from `cmex`?
- What could be added to `cmex`?
- What could be taken out of `cmex`?

The aim of this paper is to help the MFG[1] design the MX encoding as a replacement and improvement of the `cmex` encoding.

Note: Most of what is written in this paper is pure theory, and has not been applied or tested.

Acknowledgements: thanks to Alan Jeffrey and Barbara Beeton for their constructive comments, and help.

1 What is in `cmex`?

1.1 Delimiters

- Four different sizes of ( ) and extensible versions. Left and right extensible modules are '102 and '103.
- Four different sizes of [ ] and extensible versions. The extensible modules, one for the right bracket, and one for the left bracket are: '066, '067.
- Same for { and } ; the extensible module is: '076.

  Note: The extensible module here is very small, because it is added twice: once above the middle piece, and once below the middle piece. Its height is half that of the parentheses' extensible module. Interesting to see that there is only one extensible module for both the left and the right curly brace. This is because the left-right extension of a curly brace is symmetrical, unlike the parentheses for example.

[1] Math Font Group.

- Four different sizes of $\langle$ and *rangle*. No extensible version.
- Same for $\backslash$ and /. No extensible version.
- Four different sizes of $\lfloor$ and $\rfloor$ and an extensible version. Extensible modules: ’066, and ’067.
- Same for $\lceil$ and $\rceil$. Same extensible modules as the previous one.
- Glyphs in positions ’014 and ’015 are the extensible versions of the vertical bar and the double vertical bar. They are their own extensible modules.

1.2 Large operators

Large operators come in pairs:

- The sqcup
- The circled integral
- The circled dot
- The circled plus
- The circled times
- The sums
- The prods
- The normal integrals
- The bigcups
- The bigcaps
- The U plus
- The wedges
- The vees
- The coprods

1.3 Wide accents

- 3 sizes for the hat
- 3 sizes for the tilde

1.4 Radicals

- Five radical signs
- The vertical bit needed to construct the big radical: ’165
- The top bit of the constructed radical: ’166

1.5 Arrows

- The three pieces for the construction of the vertical double arrow: '167 '176 '177
- The three pieces of the vertical single arrow: '077, '170, '171

1.6 Horizontal curly braces

- The four pieces for the construction of horizontal curly braces: '172 – '175

2 TEX's behavior with `cmex` glyphs

2.1 Large operators

- A large operator is vertically centered with respect to the math axis. This means that, whatever the surrounding glyph size, things will not look too bad.
- With the following definition of a large operator: `\mathchardef \sum = "1xyy`, if TEX is in *display style*, it looks to see if the character in position `"yy` of family `x` has a successor. If it does then the successor is taken. When not in *display style*, TEX just takes character `"yy` from family `x`. Thus in text style, in script style and in scriptscript style the same glyph is used.

2.2 Vertical delimiters, and friends

Radicals are delimiters, and vertical arrows also, so I shall only speak about delimiters. Here is a quote from Victor Eijkhout's book:

> A delimiter has two codes: a small variant, and a large variant. TEX first tries the small variant, and if that is not satisfactory (or if the left part of the delimiter code is 000) it tries the large variant. If trying the large variant does not meet with success TEX takes the largest delimiter encountered in this search. If no delimiter at all is found, (which can happen if the right hand part is also 000), an empty box of width `\nulldelimiterspace` is taken.
>
> Investigating a variant means in sequence:
>
> - If the current style is scriptscript style, the scriptscript font of the family is tried.
> - If the current style is script style or smaller the script font of the family is tried.
> - Otherwise the text font of the family is tried.
>
> Looking for a delimiter at a certain position in a certain font means:
>
> - If the character is large enough, accept it.
> - If the character is extendible accept it.
> - Otherwise, if the character has a successor (the same but bigger), try the successor.

Using the three size mechanism probably did not seem necessary to Knuth. Generally large delimiters are used in display style, and not in script or scriptscript style. However, they can also be used in the small styles.

2.3 Wide accents

For the choice of accents, TEX only considers one font, but looks to see if the current accent has a successor. Unlike the delimiter choice mechanism, the accent choice mechanism does not go through all three sizes. TEX chooses the accent in such a way that the accent width is as close as possible but smaller than the width of the box to cover.

2.4 First conclusion

For operators, a distinction in size is made between text style and display style, whereas with the usual automatic size choosing mechanism, glyphs in text style and in display style are taken from the text size font, and are therefore the same size.

3 Consequences of loading `cmex` in 3 different sizes

3.1 Consequences for operators

For operators, I shall consider two completely separate policies. In one case I will consider that `cmex` is unchanged, and loaded in three sizes. The other policy is to consider that if `cmex` is loaded in three sizes, we no longer need to have two sizes of operators in `cmex`, thus `\bigcup` would not have a successor in its font. For better distinction, I shall call the latter (and nonexistent) version `cmex′`. *All the following supposes that no new macros have been written. I'm just trying to see what TEX's automatic behavior would be.*

If `cmex′` is loaded in three sizes. In such a situation TEX has a large version of `\bigcup` (and other operators) in text size, a small version in script size, and yet a smaller version in scriptscript size.

The operators can be centered: no problem.

In script and scriptscript style the resulting 'big operators' would be smaller than if they were produced with today's standard TEX, and todays standard `cmex`. What is more, one would be smaller than the other, which is also not the case with today's standard TEX, and todays standard `cmex`.

In display style one would get big operators from the text size font: that's all right. *But one would also get a big operator in text style,* and that is not conform with today's standard TEX behavior.

If the existing `cmex` encoding is loaded in three sizes. The operators can be centered: no problem.

In script style, one would get the smallest version of a large operator. But coming from a small size font, that will produce something very small. In scriptscript style, same behavior as in script style, but the result would be even smaller. Thus in script, and in scriptscript style, the large 'big operators' would never automatically be used. That is why I thought up the `cmex′` encoding.

In text style, TEX would produce the small version taken from the text size font. In display style TEX would produce the big version of operators taken from the text size font.

So in text style and in display style, there would be no change compared to what today's standard TEX produces. But script and scriptscript style would produce different results.

In both cases, things could be improved if macros were written to override the present behavior of `\bigsqcup`. I would think of things like `\mathchoice`, but ...

3.2 For vertical delimiters, radicals, vertical arrows

Let's start by supposing TEX is in scriptscript style, and it has to typeset a large delimiter. One should consider two cases:

The delimiter has an extensible variant. In this case the search will start in scriptscript size, and continue until TEX finds the extensible variant of scriptscript size. Then the search will stop, and the extensible will be used. This extensible will come from scriptscript size, and therefore probably not look the same as it would in today's setup, where all extensibles come from text size.

The delimiter does not have an extensible variant. As previously, the search starts in scriptscript size. If nothing big enough is found in scriptscript size, the search continues in script size. If still nothing is found, the search then continues in text size. If necessary the biggest delimiter from text size will be used. If the search stops in text size, there is no difference with what TEX produces today. But if the search stops before reaching text size, the chosen delimiter will be different from the one TEX would use in the present configuration. Its strokes would be finer, and better adapted for use in script style.

If one supposes that TEX is in script style, the previous two cases also apply, except that every occurrence of 'scriptscript' must be replaced with 'script'. If one supposes that TEX is in text style, the result of loading three different sizes of `cmex` would be the same as it is in TEX's current configuration.

3.3 For horizontal curly braces

If they are automatically taken from script size, or from script script size when necessary, the spacing changes a little, because the dimensions in the `.tfm` files would be different. A consequence of this could be different line and page breaks.

However, I think that it would be nice if curly braces did come out of the correctly sized fonts. Then their boldness would match the surrounding text. But I have been told that on a macro programming point of view things could be difficult, even if the glyphs are available and loaded.

3.4 For wide accents

See first paragraph of previous section.

To my taste if accents were taken from the current size, things could only look better. The accent's width would be closer to that of the material under the accent, and the accent's boldness would be better adjusted.

Note: Unlike the delimiter choice mechanism, the accent choice mechanism is restricted to one font, and one size. It will thus not look in text size when it is in script size for instance. So in script style, accents will always come out of the script size font, and in scriptscript style, accents will always come out of the scriptscript size font, etc....

3.5 Conclusion

Nearly everything in `cmex` could have lived in a normal three sized math font, and that would maybe have been better. The only problems would have come from the specific "big operator" behavior that Knuth wanted.

Also one must not forget that Knuth did not want to leave any empty slots.

The reduced amount of memory that was available on the machines with which TEX was first used could have been another reason for loading `cmex` in one size only.

4 What could be added to `cmex`?

I am now considering possible evolutions of `cmex`, although I use the terminology "adding to `cmex`", the font resulting from these evolutions would have a different name.

4.1 If the `cmex` encoded font is loaded in three sizes

In this case big operators will not produce the usual results, and the rest will be slightly different, as stated above.

- One can add wide accents, but one will get slightly different (better) results. Thus wide accents will match the script and scriptscript styles. Macros could be made (available as a style option) to keep the old behavior, if necessary.
- One can increase the number of different sizes for accents.
- One can add big delimiters and their extensible versions, without any problem! Accepting that things will be slightly better adjusted in script and scriptscript style. Macros could be made (available as a style option) to keep the old behavior, if necessary.
- One can increase the number of different sizes for delimiters, and one can probably reduce the height of the extensible module in order to make the growing of delimiters more gradual.
- One can add some vertical extensible arrows! Accepting that things will be slightly better adjusted in script and script script style.
- One can add small and large 'big operators' without any problem!
- Big improvement: one can add loads of other stuff (symbols, etc...) that would come in all three sizes.

4.2 If the `cmex` encoded font is only loaded in one size

- One can add big delimiters and their extensible versions without any problem!
- One can increase the number of different sizes for delimiters, and one could reduce the height of the extensible module in order to make the growing of delimiters more gradual.
- One can add some vertical extensible arrows!
- One can add large operators without any problem!
- One can add wide accents without any problem, and the present behavior of wide accents will not change. But if wide accents are meant to match the script and scriptscript styles, then wide accents must go in another font that would be loaded in different sizes.
- One can increase the number of different sizes for accents.
- One could add other stuff, but it would only come in one size.

4.3 If a cmex′ encoded font is loaded in three sizes

I am considering here the imaginary cmex′ encoded font, that I described previously.

One can add the same things as when cmex is loaded in three sizes. The only difference is: if no macro programming is done, the text style and display style will produce the same 'big operators'. In script and scriptscript style the 'big operators' would be in different sizes from one another and smaller from those in text style.

5 Conclusions

If one loads cmex in three different sizes, many things are improved, and with a \mathchoice the initial behavior of large operators could be kept, or available as a style option.

If cmex is kept in a single size, we must decide whether to put wide accents in or not.

6 The beginning of my cmex10.pl file

```
(FAMILY CMEX)
(FACE O 352)
(CODINGSCHEME TEX MATH EXTENSION)
(DESIGNSIZE R 10.0)
(COMMENT DESIGNSIZE IS IN POINTS)
(COMMENT OTHER SIZES ARE MULTIPLES OF DESIGNSIZE)
(CHECKSUM O 37254272422)
(FONTDIMEN
   (SLANT R 0.0)
   (SPACE R 0.0)
   (STRETCH R 0.0)
   (SHRINK R 0.0)
   (XHEIGHT R 0.430555)
   (QUAD R 1.000003)
   (EXTRASPACE R 0.0)
   (DEFAULTRULETHICKNESS R 0.039999)
   (BIGOPSPACING1 R 0.111112)
   (BIGOPSPACING2 R 0.166667)
   (BIGOPSPACING3 R 0.2)
   (BIGOPSPACING4 R 0.6)
   (BIGOPSPACING5 R 0.1)
   )
(CHARACTER O 0 ...
```

6.1 Comments about the cmex10.pl file

- The xheight is not null.
- The space is.
- With the following:

```
(CHARACTER O 100
   (CHARWD R 0.875003)
   (CHARHT R 0.039999)
   (CHARDP R 1.760019)
```

```
(VARCHAR
   (TOP O 70)
   (BOT O 73)
   (REP O 76)
   )
)
```

that is in the `.pl` file, one can produce something that looks like a growing integral:

$$\int \frac{3.q}{\frac{3\pi.r^2}{3.q.b.c}}$$

- The pieces used to construct the horizontal curly braces are not linked in any way.
- The bottom pieces of the extensible parentheses are overloaded for `\rmoustache` and `\lmoustache`. One of these could be linked (charlisted) with the integrals, so that `\left\bigint` could produce a growing integral like the delimiters.
- The bottom pieces of the curly braces ('072 and '073) are also overloaded for `\lgroup` and `\rgroup`.
- The middle pieces of the braces are overloaded for `\arrowvert` and `\Arrowvert`.
- The extensible module of the braces is overloaded for `\bracevert`.
- For the wide accents and the curly braces the depth is null.
- All the delimiter glyphs in `cmex` are set with a very small height and a big depth. This is because the radical primitive is also used for delimiters. For radicals, the `.tfm` height of the glyph is used to determine the size of the hrule.
- The extension modules do not have any height at all. Same for the arrow heads.
- The four integrals have italic corrections.
- Small versions of operators have a null height, whereas big versions have a small height and a big depth:

```
  (CHARACTER O 116
   (COMMENT This is the small \bigotimes)
   (CHARWD R 1.1111145)
   (CHARDP R 1.000013)
   (NEXTLARGER O 117)
   )
(CHARACTER O 117
   (COMMENT This is the big \bigotimes)
   (CHARWD R 1.511116)
   (CHARHT R 0.100001)
   (CHARDP R 1.500012)
   )
```

  I have no explanation for this!

- There are no kerns or ligatures in `cmex`.

7 General comments