Ideas for $\mathcal{NTS}$/ε-TeX

This list has not be sorted by any criterion!

- Some well-known shortcomings of TeXs math mode which everybody wants lifted:
  - More than 16 families.
  - A `\currentstyle` parameter. This would need the implementation of a new primitive like `\frac`. The variable would be guaranteed to have the correct value only if `\over` is avoided in the formula.
- Remove the dependence on strange glyphs positioning: Accents should be allowed to be on the baseline, radicals should not have to have huge depths. This should be implemented in a way such that the 'traditional' positioning still works. It could be implemented through additional fontdimensions, whose absence would signal the 'traditional' setup.

  I would think of an —accentheight— fontdimension for specifying the height of *all* accents in the font if it is different from the —x-height—, and a —radical-rulethickness— to be used instead of the height of the radical glyph; the radical glyph would of course be shifted to this height.

  If this solution turns out to be too coarse, one would have to find a way to attach additonal dimensions to individual glyphs. The `\skewchar` approach is inadvisable in the long run, since it eats up valuable slots in the font. A revised —tfm— format (clearly not an ε-TeX task!) should probably allow to attach arbitrary properties to glyphs.
- Some of the fontdimensions in appendix G are used for more than one purpose, making changes difficult. These could be unbundled. Upon loading a math font, ε-TeX would check for the additional fontdimensions and if it doesn't find them, use the corresponding value from the standard set of fontdimensions.

This needs more detail!!!

- Horizontal analogues of varchar recipes. This would not break any existing document. Since existing fonts simply do not have such beasts. ε-TeX should do the right thing when the last character in the charlist for an math accent is an extensible glyph.

  This would make it easier to have many sizes of horizontal parens, braces, brackets, etc and might also make the implementation of the `\overbrace` and `\underbrace` macros easier.

  This is one of the cases where DSSSL has a flow object class for math (marks) which is not directly supported by primitives of TeX.
- A primitive for underaccents, since this is almost impossible to do right in macros (at least I didn't succeed). This would need some decisions about where to store the needed metric information. I think this would need two additional dimensions: the skew, and one additional parameter to position accent parameters vertically. My proposal would be: Use ⟨*skewchar*⟩-⟨*accentee*⟩ kerning for the former and ⟨*accent*⟩-⟨*accent*⟩ kerning or ⟨*accent*⟩ italic correction for the latter.

  The design of this primitive must go hand in hand with generalizing the `\mathaccent` primitive. They should be designed so that `\mathaccent` and `\mathunderaccent` can use the same glyphs.
- Adding support for nested accents to `\mathaccent` (or a new primitive) would also be very nice, since the macros for this tend to be complicated and slow. It would be enough if `\mathaccent` would not only check for a single character but also for a (possibly nested sequence of) `\mathaccent`(s or `\mathunderaccents`) and determine the skew from the innermost accentee.
- It would also be nice to lift the '15 nonzero heights and depths' restriction from the —tfm— format, but that is clearly out of scope for ε-TeX.
- Should extensible operators like growing integrals be supported (some fonts have the necessary glyphs) ? Support for extensible operators would imply that there needs to be a way to specify that a symbol is at the same time extensible and an operator. One question which arises then is how to determine the subformula which should be covered by the operator. (In DSSSL-speak: The operator flow object really has four ports: one for sub- and superscript, one for the operator glyph itself and one for the operand. TeXs `\mathop`s do have only the first three.)

  If this is wanted, we probably need a 'diagonal variant' of extensible characters too, for slanted integrals. The parts of the extensible characters should have enough unused dimensions to allow for the specification of the slant (e. g. the italic correction of the repeatable piece).

  Diagonally extensible characters would probably have lots of other uses.

- Should there be support for left superscripts and subscripts ? This would probably be a major change, since it would imply changing the design of math lists to have more than two script fields. DSSSL has a flow object class (script) that allows six different scripts to be attached to a math atom: sub, sup, pre-sub, pre-sup, mid-sub and mid-sup.

  Proper left script positioning might also require a left analogue of italic correction, i. e. changing the —tfm— format. A left italic correction might also be needed in implementing *automatic* italic correction.