The state of OpenType math typesetting

Ulrik Vieth
Stuttgart, Germany

EuroBachoTEX 2011
Brodnica, Poland, 2011

Overview of this talk

- Review of OpenType math support
  - Technology review: Unicode and OpenType math
  - Available support in engines, macros, fonts
- Experiences with OpenType math support
  - Known problems in engines, macros, fonts
  - Workarounds and Solutions
- Samples of OpenType math typesetting
  - Choices of test platforms
  - Choices of OTF math fonts
  - Examples of math typesetting
- Summary and Conclusions

Technology Review: Unicode math

- What ist Unicode?
  - encoding standard for (input) characters and symbols
- What is Unicode math?
  - addition of math symbols and alphabets to Unicode
  - coordinated by STIX group of publishers (late 1990s)
- What does Unicode math provide?
  - hundreds of math symbols added to slots U+2xxx
  - dozens of math alphabets added to slots U+1Dxxx
  - size variants of math symbols are *not* encoded: different size, same symbol ⇒ same meaning
  - font variants of math alphabets *are* encoded: different font, same letter ⇒ different meaning

Technology Review: OpenType math

- What ist OpenType?
  - font technology for (output) glyphs and symbols
  - developed by Adobe (PostScript) and MS (TrueType)
- What is OpenType math?
  - addition of MATH table to OpenType font format
  - developed for MS Office 2007 (experimental)
- What does OpenType math provide?
  - global font metrics for spacing of math formulas
  - glyph metrics for positioning of math accents
  - lookups for horizontal/vertical variants/constructions
  - base glyphs addressed by Unicode slots
  - variant glyphs addressed through lookups

Engine support for OpenType math

- MS Office 2007, 2010
  - reference implementation of OpenType math
  - some generalizations of concepts from TEX
- XeTEX
  - support for OpenType math since XeTEX 0.97 (2007)
  - uses mapping of OpenType to TEX parameters
  - makes limited use of OpenType math features
- LuaTEX
  - support for OpenType math since LuaTEX 0.40 (2009)
  - uses combined set of OpenType and TEX parameters
  - aims to provide full support of OpenType math features

Macro support for OpenType math

- XeTeX or LuaTeX (generic)
  - only primitive support for OpenType math
  - no high-level support for OpenType math
  - `luaotfload` for low-level font loading (LuaTeX)
- XeLaTeX or LuaLaTeX
  - `fontspec` for high-level text font selection
  - `unicode-math` for math font selection
  - `expl3` for intermediate macro layers
  - `luaotfload` for low-level font loading (LuaTeX)
- ConTeXt Mk IV
  - support for OpenType math since LuaTeX 0.40
  - `font-otf.lua` for low-level font loading

Font support for OpenType math (I)

Fonts (more or less) ready for production:

- Cambria Math
  - shipped with MS Office 2007, 2010, Windows 7
  - reference implementation of OpenType math font
- Asana Math (Apostolos Syropoulos)
  - Palatino-like font derived from `pxfonts`
  - available from CTAN, included in TeX Live 2010
- XITS Math (Khaled Hosny)
  - Times-like font derived from STIX Fonts 1.0
  - available from CTAN, included in TeX Live 2010
- Latin Modern Virtual (Hans Hagen)
  - virtual OpenType font, only for ConTeXt Mk IV

Font support for OpenType math (II)

Fonts under development:

- Neo Euler (Hermann Zapf, Khaled Hosny)
  - derived from redesign of AMS Euler
  - development supported by DANTE
- Lucida Math (Bigelow & Holmes, Khaled Hosny)
  - derived from redesign of Lucida family
  - development supported by TUG
- Latin Modern, TEX Gyre Math (GUST Foundry)
  - developed with MetaPost / MetaType 1
  - development supported by various LUGs

How to use OpenType math (I)?

- ConTEXt Mk IV
  - support for OpenType math in ConTEXt kernel
  - predefined typescripts for Cambria, Asana, XITS, etc.
  - additional typescripts can be loaded as needed

```
\usetypescript[cambria]
\setupbodyfont[cambria,10pt]
```

- LuaLaTEX + XeLaTEX
  - support by `fontspec` + `unicode-math` packages
  - no predefined font sets, fonts loaded on demand
  - configuration options to customize math style, etc.

```
\documentclass{article}
\usepackage{fontspec,unicode-math}
\setromanfont[Ligatures=TeX]{Cambria}
\setmathfont[math-style=TeX]{Cambria Math}
```

How to use OpenType math (II)?

- 3 choices of TEX engines + macro packages
  - LuaTEX with ConTEXt
  - LuaTEX with LuaLaTEX
  - XeTEX with XeLaTEX
- 4 choices of OpenType text + math fonts
  - Cambria + Cambria Math
  - XITS + XITS Math
  - TG Pagella + Asana Math
  - TG Pagella + Neo Euler
- more choices of fonts to come
  - Lucida Bright + Lucida Math
  - Latin Modern + LM Math
  - TeX Gyre + TG Math

Experiences with OpenType math

- Experiences testing OpenType math
  - Testing of OpenType math = Testing of a complex system
- Possible causes of testing problems
  - Problems with TEX engines
  - Problems with macro packages
  - Problems with OpenType fonts
  - Font-loading problems

Problems with TEX engines (I)

Fatal engine problems (crashes)

- XeTEX: 64-bit binaries crashing with segfaults
  - Status: unresolved (broken) in TEX Live 2010
  - Workaround: use 32-bit binaries instead
  - Solution: patch available, wait for TEX Live 2011
- LuaTEX: crashing when loading empty fonts
  - Status: fixed in LuaTEX 0.61
  - Solution: update LuaTEX from TL Contrib
- LuaTEX: crashing when calling `os.execute`
  - Status: fixed in LuaTEX 0.70
  - Workaround: `\directlua{os.execute = nil}`
  - Solution: update TEX Live packages frequently

Problems with TEX engines (II)

Non-fatal engine problems (mis-features)

- LuaTEX: Incorrect size of big delimiters
  - Problem: off-by-one error (skipped last size)
  - Status: fixed in LuaTEX 0.61
  - Solution: update LuaTEX from TL Contrib
- XeTEX: Inconsistent alignment of super-/subscripts
  - Problem: shift depends on ascenders/descenders
  - Status: unresolved in XeTEX engine
  - Workaround: add empty groups in math input
- XeTEX: Incorrect size of wide math accents
  - Problem: width of accents includes super-/subscripts
  - Status: unresolved in XeTEX engine
  - Workaround: add empty groups in math input

Problems with macro packages

Macro problems (mis-features)

- `unicode-math`: Incorrect font metrics with XeTeX
  - Problem: OpenType math fonts loaded into family 4 but font metrics in XeTeX taken from families 2+3
  - Workaround: add explicit font loading
  - Status: fixed in recent versions of `unicode-math`
  - Solution: update TeX Live packages
- `luaotfload`: Incorrect mapping of font weights
  - Problem: demibold not recognized as bold series
  - Workaround: add explicit font names for BoldFont
  - Status: fixed in recent versions
  - Solution: update TeX Live packages

Problems with OpenType fonts (I)

Font parameter issues (mis-features)

- Cambria Math: Incorrect size of display operators
  - Problem: Incorrect value of DisplayOperatorMinHeight
  - Workaround: modify parameter value on font loading
  - Status: fixed in ConTEXt, but not in `luaotfload`
  - Solution: adopt existing fix from `font-pat.lua`
- Asana Math: Incorrect size of display operators
  - Status: fixed in recent versions
  - Solution: update TEX Live packages
- XITS Math: Incorrect values of some math parameters
  - Status: fixed in recent versions
  - Solution: update TEX Live packages

Problems with OpenType fonts (II)

Font encoding issues

- Incorrect shape of partial sign (upright vs. italic)
  - Problem: Inconsistencies in Unicode font tables
  - Status: correct in some fonts, incorrect in others
  - Solution: unlikely to be fixed anytime soon
- Inconsistent symbol and alphabet coverage
  - Problem: different fonts provide different subsets
    - some math alphabets missing or incomplete
    - missing alphabets may be substituted or disappear
    - some symbols missing in some fonts or alphabets
    - missing symbols may disappear without trace
  - Status: correct in some fonts, incorrect in others
  - Solution: check your documents and log files

Problems with OpenType fonts (III)

Font loading issues

- Font loading in XeTeX:
  - uses `fontconfig` library to locate OpenType fonts
  - may need to edit `fonts.conf` to add `texmf-local` tree
  - may need to run `fc-cache` to refresh font cache
- Font loading in LuaTeX:
  - uses `kpathsea` library to locate OpenType fonts
  - uses `fonts/opentype` to load `*.otf` fonts (e.g. Euler)
  - uses `fonts/truetype` to load `*.ttc` fonts (e.g. Cambria)
  - uses Lua-based font cache (`luaotfload`)
- Font loading in ConTeXt:
  - does not use `fontconfig` or `kpathsea` libraries
  - uses Lua-based file cache and font cache (`luatools`)
  - uses `fonts.conf` to locate font path of system fonts

Examples of testing OpenType math

- 3 choices of TEX engines + macro packages
  - LuaTEX with ConTEXt
  - LuaTEX with LuaLaTEX
  - XeTEX with XeLaTEX
- 2 choices of comparing the quality
  - Comparing LuaLaTEX vs. ConTEXt
  - Comparing LuaLaTEX vs. XeLaTEX
- 4-5 choices of OpenType text + math fonts
  - Cambria + Cambria Math
  - XITS + XITS Math
  - TG Pagella + Asana Math
  - TG Pagella + Neo Euler
  - Lucida Bright + Lucida Math

Comparing the quality of OpenType math

- Comparing LuaLaTeX vs. ConTeXt:
  - different user interface (`unicode-math` vs. ConTeXt)
  - similar font loading code (`luaotfload` vs. ConTeXt)
  - same underlying TeX engine (LuaTeX in both cases)
  - same implementation of math algorithms
- Expectations
  - similar input expected to produce similar output
  - BUT: cannot use identical input for test document
  - primarily used for verification of bugs/features

Comparing the quality of OpenType math

- Comparing LuaLaTeX vs. XeLaTeX:
  - same user interface (`unicode-math` in both cases)
  - different font loading code (`luaotfload` vs. XeTeX)
  - different underlying TeX engines (LuaTeX vs. XeTeX)
  - different implementations of math algorithms
- Expectations
  - same input expected to produce different output
  - BUT: same input can be processed unchanged
  - primarily used for discovery and analysis

Methods of testing OpenType math (I)

- Systematic Testing
  - not enough time to do systematic testing
  - too much tedious work needed for full test
  - too many possibilities / combinations to test
- Sampling vs. Testing
  - create test document for sampling of notations
  - typeset same test document with each font
  - typeset same test document with each engine
  - inspect and compare test results
- Coverage Testing
  - create test document for all math alphabets
  - typeset same test document with each font
  - inspect for missing alphabets or symbols

Methods of testing OpenType math (II)

- Analysis of test results
  - inspect results of test documents for bugs
  - compare results of test documents for quality
- Analyzing large-scale effects
  - large-scale effects easy to find by visual inspection
  - usually caused by problems in engines, fonts, macros
  - usually possible to avoid or fix by workarounds
- Analyzing small-scale effects
  - small-scale effects only visible after fixing large-scale
  - effects can be highlighted in multi-color overlays
  - some effects caused by problems in fonts or macros
  - some effects expected due to engine differences

Results of testing OpenType math (I)

- Original goal
  - study effects on quality of different implementations
  - XeTeX makes only limited use of OpenType math
  - LuaTeX provides full support of OpenType math
- Observations
  - most large-scale effects caused by bugs (unintentional)
  - remaining small-scale effects smaller than expected
- Next Steps
  - remaining effects caused by engine differences
  - remaining effects need further detailed study
  - results of comparison remain inconclusive for now

Results of testing OpenType math (II)

- Additional goal
  - find out how well OpenType math support works
- Observations
  - many problems found during TEX Live 2010 pretest
  - many problems resolved shortly after release
  - few problems still remain unresolved for now
- Conclusions
  - LuaTEX is well supported and has fewer known bugs
  - XeTEX is essentially unsupported, bugs unresolved
  - macro support is equivalent for both engines now
  - macro packages are generally well supported
  - font support varies, some fonts are well supported
  - OpenType math is “bleeding edge” with all its risks