IAMC and MathML Generation

Paul S. Wang
Institute for Computational Mathematics
Kent State University
http://horse.mcs.kent.edu/~pwang

Contents

- Math Communication on the Web/Internet
- Some Examples and Live Demo
- Standards for Math Encoding
- MathML Presentation Codes
- MathML Content Codes
- Automatic MathML Generation
- Further Work

Math on the Web

- Table of integrals — mathematical database at the U. C. Berkeley.
- Live computation demos — derivatives, polynomial factoring, Fortran code generation, curve/surface plotting on `SymbolicNet` at ICM/Kent.
- *Techexplorer* — a Web browser plug-in that dynamically formats and displays documents containing scientific and mathematical expressions coded in TEX/LATEX by IBM Watson Research Center.
- NetSolve — a system to make numerical computation packages available to Web through a Java Applet by a joint project between the U. of Tennessee and the Oak Ridge National Laboratory.

Representation Standards

- MathML — a language for markup of mathematical expressions by a group at W3 consortium.
- OpenMath — a char-based math expression encoding format by the OpenMath group.
- MP — a binary mathematical expression encoding format and transfer protocol by the MP group.

Design Goals of MathML

MathML has been designed with the following ultimate goals:

- encode math for teaching and scientific communication at all levels
- encode both notation and meaning
- facilitate conversion to and from other math formats Important output formats:
  - graphical displays
  - speech synthesizers
  - computer algebra systems' input
  - other math layout languages, such as TeX

- plain text displays, e.g. VT100 emulators
- print media, including braille

- allow the passing of information intended for specific renderers and applications.
- support efficient browsing for lengthy expressions
- provide for extensibility
- be well suited to template and other math editing techniques
- be human legible, and simple for software to generate and process

Implementation Considerations

MathML rendering and processing software functionalities:

- MathML equations in HTML pages should render properly in popular Web browsers, in accordance with reader and author viewing preferences, and at the highest quality possible given the capabilities of the platform.
- HTML documents containing MathML equations should print properly and at high-quality printer resolutions.
- MathML equations in Web pages should be able to react to mouse gestures, and coordinate communication with other applications through the browser.

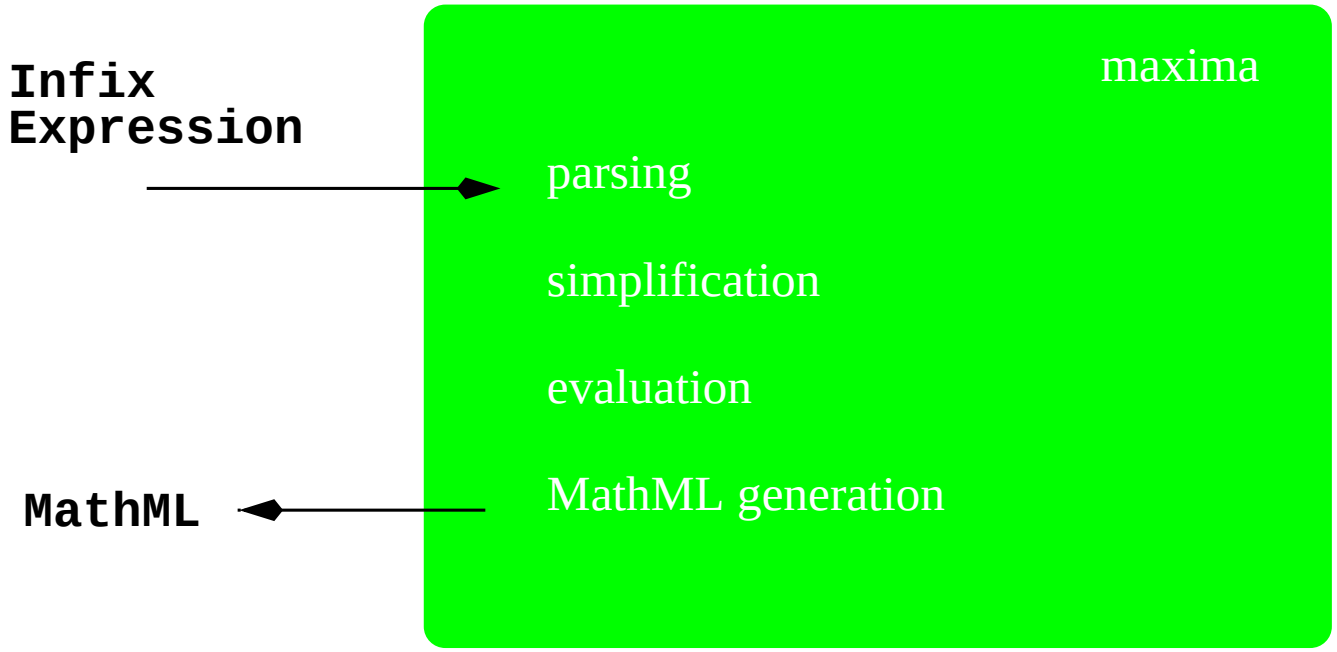
- Equation editors and converters should be developed to facilitate the creation of Web pages containing MathML equations.

MathML Systems

- Amaya — experimental browser from W3C that supports MathML
- WebEQ — a suite of Java programs from Geometry Technologies Inc. for creating and displaying Web documents containing mathematical formulas
- Techexplorer — a Web browser plugin form IBM for displaying TeX/LaTeX and MathML
- EzMath — a visual editor that helps convert widely used mathematical notations to markup codes, in either EzMath or MathML, ready for inclusion in Web pages.

Further Info on MathML

- MathML main page at W3C `www.w3.org/TR/PR-math/`
- MathML Version 1.01 specification
  `www.w3.org/1999/07/REC-MathML-19990707`
- IAMC homepage
  `icm.mcs.kent.edu/icm/research/iamc.html`
- MathML resource page
  `www.webeq.com/mathml/resources.html`


Infix
Expression
maxima
parsing
simplification
evaluation
MathML generation
MathML

Input to Maxima : `4*x - sin(%pi*x/2);`

Maxima display:

```
                                %PI X
(D5)                 4 X - SIN(-----)
                                  2
```

Parsed input:

```
((MPLUS) ((MTIMES) 4 $X)
         ((MMINUS) ((%SIN)
             ((MQUOTIENT) ((MTIMES) $%PI $X)
               2))))
```

Simplified input:

```
((MPLUS SIMP)
   ((MTIMES SIMP) 4 $X)
     ((MTIMES SIMP) -1 ((%SIN SIMP)
     ((MTIMES SIMP)
        ((RAT SIMP) 1 2) $%PI $X))))
```

Presentation Mathml:

```
<math> <mrow>
  <mrow>
    <mn>4</mn><mo>&InvisibleTimes;</mo><mi>X</mi>
  </mrow>
  <mo>-</mo><mi>sin</mi><mo>&ApplyFunction;</mo>
  <mrow>
    <mo>(</mo><mfrac>
    <mrow>
       <mi>&pi;</mi><mo>&InvisibleTimes;</mo>
             <mi>X</mi>
    </mrow>
    <mn>2</mn></mfrac><mo>)</mo>
  </mrow>
</mrow> </math>
```

Presentation Mathml:

```
<math> <apply> <plus/>
  <apply> <times/>
    <cn type="integer">4</cn><ci>X</ci>
  </apply>
  <apply> <minus/>
    <apply> <sin/> <apply> <times/>
       <cn type="rational">1<sep/>2</cn>
       <ci type="constant">&pi;</ci><ci>X</ci>
    </apply> </apply>
  </apply>
</apply> </math>
```

Conversion Algorithm

1. If expression is an single token, call `pr_token`
2. If expression is an array, call `pr_array`
3. If expression is in parenthesis, call `pr_paren`
4. If leading operator is known, call its pr function.
5. If leading operation is some other function or operator, call pr for general function

The SQRT Function

```
(defun mPr-sqrt (mexpress)
  (tprinc "<msqrt>")
    (mPr_engine (cadr mexpress)
                  'mparen 'mparen)
  (tprinc "</msqrt>")
)
```

Infix Conversion

Certain operators such as times, plus, and, or, greater-than, ... requires prefix to infix conversion.

```
(defun mPr-plus (mexpress)
  (let ((moperands (cdr mexpress))
        (flag_trunc (member 'trunc
           (car mexpress) :test #'eq)))
    (cond
     ((equal (length moperands) 1)
        (mPr-prefix mexpress))
     (t (row-begin "<mrow>")
        (mPr_engine (car moperands) lop 'mplus)
        (print_op_oprd (cadr moperands))
        (mPr-plus1 (cddr moperands) flag_trunc)
        (row-end "</mrow>")
))))
```

Further Work

- Testing and debugging
- Conforming to MathML 1.01
- Writing MP <--> MathML converters
- Writing MathML parser