José Grimm
Producing MathML with Tralics

In: Petr Sojka (ed.): Towards a Digital Mathematics Library. Paris, France, July 7-8th, 2010. Masaryk University Press, Brno, Czech Republic, 2010. pp. 105--117.

Persistent URL: http://dml.cz/dmlcz/702579

# Producing MathML with Tralics

José Grimm

Institut National de Recherche en Informatique et en Automatique
2004 Route des Lucioles
06902 Sophia Antipolis (France)
Jose.Grimm@sophia.inria.fr

**Abstract.** We describe here how Tralics can be used to convert LaTeX documents into XML or HTML. It uses an ad-hoc DTD (a simplification of the TEI), but the translation of the math formulas is conforming to the presentation MathML 2.0 recommendations. We explain how to run and parametrize the software. We give an overview of the various MathML constructs, and how they are rendered by different browsers.

## 1 Introduction

Tralics is a free software, written by the author, that converts LaTeX documents into XML; initially designed for the Inria's annual activity report, it is used by CEDRAM [1] or Zentralblatt MATH. It can be downloaded from the Web[1], where you can find some documentation and examples[2].

The software uses the same rules as TeX for analyzing the source document, and converts it into a list of tokens; these tokens may be stored into token lists or macros, as in TeX, or converted into XML elements (which play the some role as boxes in TeX). The whole XML tree is dumped into a file at the end of the run (after producing an index, a bibliography, and checking that labels are correct). The structure of this tree is inspired by the TEI DTD, for instance, a footnote is translated as a `<note>` element, with an attribute pair *place='foot'*. It is possible to change the names of almost all elements or attributes, either in a configuration file, or in the TeX source, and Tralics makes no attempt to check that the XML tree is conforming to the DTD declared in the header.

Tralics is one of the five converters examined in [6], which states some strengths and weaknesses. The authors indicate a success rate of only about two percent, and an average of 50 undefined commands per document. There are, in effect, frequently used commands (for instance `\author`, `\email`), that are defined in class or style files, not yet handled by Tralics. This is a minor inconvenience, since it is rather easy to define these commands in order to produce a `<author>` or `<email>` element. Some very useful packages like *xypic*, *pgf*, *pstricks* are unknown to Tralics. These are big and complicated packages, and all assume that there is a PostScript-like renderer that will draw the lines,

[1] `http://www-sop.inria.fr/apics/tralics`
[2] see also `http://www-sop.inria.fr/apics/gaia`

Petr Sojka (editor): DML 2010, Towards a Digital Mathematics Library, pp. 105–117.
 ISBN 978-80-210-5242-0

arrows, etc. This is incompatible with the HTML model. One solution would be to use SVG for the graphics, but the W3C example mixing SVG and MathML does not render correctly on my machine.

On the other hand, Tralics is designed to run in batch mode. In case of errors, a message is printed in the transcript file, and translation continues (the job is aborted after 5,000 errors); in order to help debug, each error produces an `<error>` element in the XML file. For instance, the translation of `\foo\bar` on line 13 will be

```
<error n='\foo' l='13' c='Undefined command'/>
<error n='\bar' l='13' c='Mathonly command'/>
```

## 2 Running Tralics

Normally, you start with a source file, say *foo.tex*, and run Tralics, this gives you a file *foo.xml*. There is an interactive mode, where Tralics reads characters from the terminal, and prints all MathML expressions on the screen. There are many ways to parameterize the output of Tralics. First of all, Tralics reads the file *foo.ult*; you can put here some LaTeX definitions (that are not used when you compile the file with LaTeX). If your document uses a class, say *c1.cls*, Tralics reads *c1.clt* instead, and if your document uses a package, say *p2.sty*, Tralics reads *p2.plt* instead; if these files are missing, they are silently ignored. Only standard classes and packages have an associated *clt* or *plt* file.

In [5], we explain how the XML documents produced by Tralics can be converted into a sequence of HTML pages, via an XSLT processor, lots of style sheets, and an external tool that converts non-HTML pieces (syntax trees and math formulas) into images (an example is Inria's activity report[3], where passiveTeX converts XML to PDF, [2]). Reasons for converting math formulas into images include the fact that few browsers render MathML, that they require specific DTDs or file names, and that the rendering is not always legible. In this paper, we assume on the contrary that the document will be interpreted by a MathML-aware browser.

Editors like CEDRAM or Zentralblatt use it to convert the meta-data (typically the summary) of a paper into XML, and insert this into an HTML page with other stuff; in Figure 1 you can see a MathML formula in an article title (more complex formulas appear in the abstracts, or reviews; some are in display style).

There is a special feature in Tralics in that a math formula can be translated either as a LaTeX-like formula or a MathML formula; the following defines a command `\both` with one argument that evaluates it in both modes (this feature is used by CEDRAM), and puts the result in an element named A and B respectively.

```
\def\Va#1{\xbox{A}{#1}}
```

[3] `http://www.inria.fr/rapportsactivite/index.fr.html`

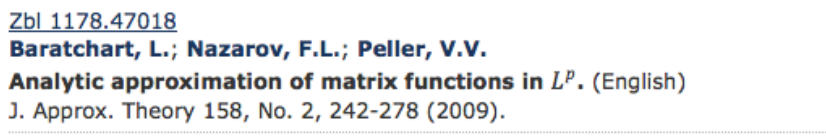


**Fig. 1.** Example of MathML in Zentralblatt

```
\def\Vb#1{\xbox{B}{\@nomathml=-1 #1}}
\def\both{\@reevaluate\Va\Vb}
```

The translation of

```
\def\square#1{(#1)^2}
\both{$\square{a+b}$}
```

will be

```
<A>
 <formula type='inline'>
  <math xmlns='http://www.w3.org/1998/Math/MathML'>
    <msup>
      <mrow>
        <mo>(</mo>
        <mi>a</mi>
        <mo>+</mo>
        <mi>b</mi>
        <mo>)</mo>
      </mrow>
      <mn>2</mn>
    </msup>
  </math>
 </formula>
</A>
<B><texmath textype='inline' type='inline'>(a+b)^2</texmath></B>
```

You can change the name of the elements or attributes used by Tralics, either on the command line, or in a configuration file, or in the source file; this applies to almost everything but the content of math formulas (the content of the `<math>` element is always conforming to the MathML recommendations). For instance, if you say

```
\ChangeElementName*{mathmlns}{foo}
\ChangeElementName{math}{mml:math}
\ChangeElementName{formula}{F}
\ChangeElementName*{type}{T} ...
```

this will change the name of the elements `<math>` and `<formula>`, or of the attributes *type*, etc. With these declarations, the translation of `$x$` will be

```
<F T='I'>
  <mml:math xmlns='foo'>
    <mi>x</mi>
  </mml:math>
</F>
```

All math formulas are wrapped into a `<formula>` element that has various attributes: the name of the environment (if any), the value of the equation number (value of `\theequation`), unique ID for referencing, tags, etc. Note that at most one label can be attached to a formula (this limitation will be removed in the future); but using multiple tags is problematic, because of the bad rendering of the `<mlabeledtr>` element by most browsers. For this reason, the translation of the `\tag` command can be either a part of the math expression, or an attribute of the formula. The translation of

```
\begin{equation} x\end{equation}
\begin{equation*} x \tag{1-2}\end{equation*}
\begin{math} x\end{math}
```

will be

```
<F id-text='1' id='uid1' textype='equation' T='D'>
  <mml:math mode='display' xmlns='foo'> ...
  </mml:math>
</F>
<F textype='equation*' T='display' tag='(1-2)'>
  <mml:math mode='display' xmlns='foo'> ...
  </mml:math>
</F>
<F textype='math' T='I'><mml:math xmlns='foo'> ... </mml:math></F>
```

## 3 Examples

We shall show in this section some screen shots; they were obtained using different browsers, essentially Amaya and Firefox on MacIntosh. We carefully followed the instructions for installing the right fonts in 2007: we used TEX's Computer Modern fonts, set the user preferences (asking Firefox to use CM fonts) on Linux, and installed the Mathematica 4.1 fonts instead on the Mac. Our HTML example file (containing all formulas found in the TEXbook or

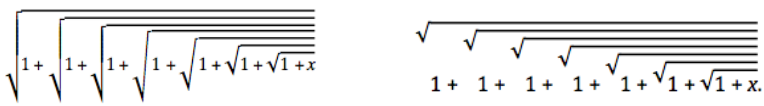

**Fig. 2.** Nested roots (Firefox 2007 & 2010))

LaTeX companion, [4]) was converted to PDF (via the Print button) and saved somewhere. Since then, the machines have been replaced by new ones, and the snapshots were taken with default settings. The Firefox web page recommends to use the STIX fonts (released May 28, 2010), and to reset the user preference item. This has no effect on Amaya or Opera, but corrects a few bugs for Firefox.

We show on Figure 2 the rendering of nested square roots. Defaults fonts were used on the right; installing Stix fonts improves the situation on Mac (but not on Linux). Consider another example

```
\[ \left(\left[\left\lbrack a\left\{\left\lbrace\left\lfloor b
 ...
\rbrack\right]\right)\]
```

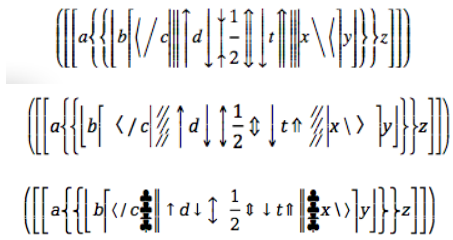

**Fig. 3.** Various delimiters (Firefox Mac 2007, Mac 2010, Linux 2010)

The rendering is on Figure 3. There was a mistake in the source of the 2007 edition of the document (there was `\langle` instead of `\rangle`). The bugs have been corrected in the Stix fonts.

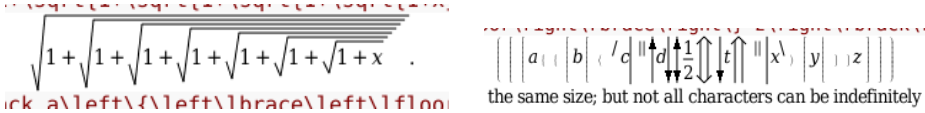


**Fig. 4.** Delimiters and Roots (Amaya)

We show on Figure 4 the rendering by Amaya. One can notice that the math formula is very near to the surrounding text. The square root signs are beautiful; the delimiters are unreadable: the parentheses, brackets and vertical lines all look the same; the forward slash, backward slash and double vertical rule are not vertically centered; braces are much too small. Magnifying the font does not help distinguish ( from [.

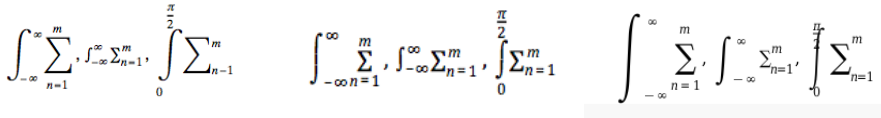

**Fig. 5.** Large operators (Firefox 2007, Firefox 2010, Amaya)

We show on Figure 5 the placement of indices for large operators (a sum and an integral) in display style, in text style, and in display style with non-standard placement. Note that the new Firefox uses the same sum for all three occurrences (whether or not Stix fonts are installed), while the sizes of the integrals change; the placement of the last $\pi/2$ is wrong in Amaya.

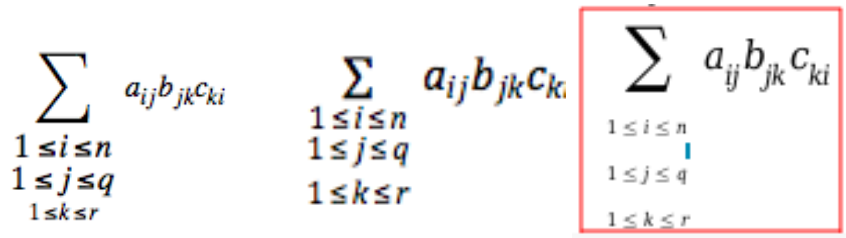

**Fig. 6.** Stacking (Firefox 2007, Firefox 2010, Amaya)

Figure 6 corresponds to the following example

```
$\displaystyle \sum\limits_
  {{\scriptstyle1\le i\le n\atop\scriptstyle 1\le
    j\le q}\atop \scriptstyle 1\le k\le r}
 a_{ij}b_{jk}c_{ki}$
```

The math formula consists of a `<mstyle>` element, so that the formula should be rendered in display style; this means that a large sum sign should be used. The sum is followed by three `<msub>` elements, letters a, b and c should have a normal size, letters i, j and k should be smaller. The index in the sum is special; Tralics chooses to use `<munder>` in display style. The index is formed of three lines, one atop the other, and we use a `<mfrac>` with zero line thickness. Each line has its style that says that i, j and k should have the same size everywhere. They are too small in Amaya, and sometimes too big in the old Firefox version. Note that the vertical spacing is irregular in Firefox, too big in Amaya. We have also shown the bounding box in the Amaya case. The rendering of MathML in Firefox and Amaya seems quite good, contrarily to Opera: Figure 7 shows that some parts are correctly rendered (the square roots), some are quite wrong (the middle part should be the same as in Figure 3), and some are partially wrong (the fraction rule should be invisible on the right).

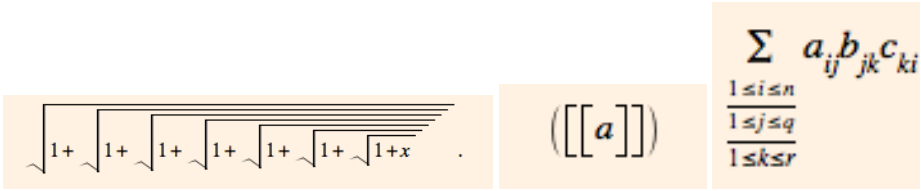

**Fig. 7.** Examples with Opera (version 10.10)

## 4 The Characters

When Tralics sees a dollar sign, or any other command that indicates the start of a math formula, it reads all characters up to the end of the formula, expanding what can be expanded, evaluating what can be evaluated, sometimes signaling errors. The result is a nested list of math tokens, and a two-pass algorithm is used to convert this into a MathML element. Example

```
$ \{x,~x^{10} > \alpha\beta\} \cup \left[ xy\right]$
```

The first pass converts atoms into atoms, the second pass recursively combines these objects. To each of the three tokens `\alpha`, `\{`, and `x` is associated a class and a value. In TEX, the numbers associated to the first two tokens are 010B or 4266308, the class is respectively 4 or 0, the value is either 10B or the pair (266, 308) for `\{`; a pair is required here since the glyphs associated to the brace are in two different fonts (typically cmsy and cmex); this is a strange design. Note that there are at most $2^{12} = 4096$ different possible values, which is a very small number. For us, the value will be any Unicode code point. The objective of the first pass is to replace these numerics value by atomic XML elements (called token elements), leaving the class unchanged. The second pass builds the tree looking only at the classes.

The translation of an atom depends on the class, it is a `<mi>`, `<mn>`, `<mspace>`, `<mo>`, if this atom is a letter, a digit, space, or something else; in some cases a sequence of atoms is converted to a single token element. A simplified representation of the previous formula could be "liosiC(nn)oiiro(ii)", where the meaning of i, n, s and o, is clear, parentheses indicate nesting, C indicates a command interpreted later, l and r indicate opening and closing delimiters.

A Unicode code point (see [7]) is integer less than $17 \times 2^{16}$. Each sequence of $2^{16}$ characters is called a plane; the first plane contains all European characters, a lot of mathematical characters, and browsers like Firefox have glyphs for all characters in this plane. Any positive number that fits on 27 bits is considered by Tralics as a character, thus `$\char"7FFFFF$` is accepted, but produces an illegal Unicode character. If you specify UTF-8 as current input encoding, then the sequence of 3 bytes E2 82 AC will be read as the character U+20AC; it can also be entered as `^^^^20ac`; it corresponds to the Euro sign. Any positive number that fits on 16 bits is a normal character (it has a category code, a UC code, can be part of a command name, etc.), it can be entered via the four-hat

mechanism, is part of the first plane. Note that TEX extensions like LuaTEX consider any Unicode character as a normal character, that can be entered via a five-hat or six-hat mechanism.

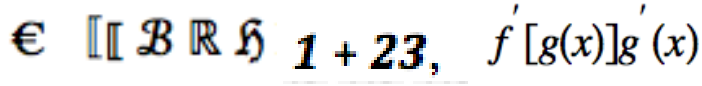

**Fig. 8.** Examples of characters

Consider for instance the characters `^^^^301a` or `^^^^27e6`; both are named "left white square bracket" by the Unicode consortium, the latter one being qualified as "mathematical", they look quite the same in Firefox (see Figure 8). When used in math mode, the translation is a `<mi>` element; you can use the commands `\mathco` or `\mathci` if you want a `<mo>` or `<ci>` instead (there are six possibilities). One can note that the prime sign is positioned too high, especially for the letter g (this is because the prime sign is an active character in math mode and is converted into a superscript).

At position 1Dxxx (in the second Unicode plane), you can find a lot of variants of mathematical characters (bold, Fraktur, bold Fraktur, double struck, etc.), but some characters appear in plane one (see Figure 8). A complete set of fonts (like the Stix fonts) is required in order the see them all.

Most useful characters can be used in Tralics via a name, for instance `\texteuro` produces the Euro sign. The command `\llbracket` can be used to produce ⟦. The translation is

```
<mo>&LeftDoubleBracket;</mo>
```

The entity used in the previous formula is defined in the file *mmlalias.ent* (distributed with the MathML DTD, and Tralics). You can easily convince Tralics to use the character U+301A instead of the entity.

The translation of the formula is

```
<mrow>
  <mrow>
    <mo>&#x0007B;</mo><mi>x</mi><mo>,</mo>
    <mspace width='3.33333pt'/>
    <msup><mi>x</mi> <mn>10</mn> </msup><mo>&gt;</mo>
    <mi>&#x3B1;</mi><mi>&#x3B2;</mi><mo>&#x0007D;</mo>
  </mrow>
  <mo>&#x0222A;</mo>
  <mfenced separators='' open='[' close=']'>
    <mi>x</mi><mi>y</mi></mfenced>
</mrow>
```

You can notice in this example that Tralics uses a single <mn> for the whole number, and considers $xy$ as two consecutive identifiers. Any math formula that contains at least two objects is enclosed in a <mrow> or a <mfenced> element (we make the assumption that <mrow> behaves the same as a <mfenced> without separators, with empty open and closing delimiters). When the <mfenced> element has only one item, the *separators* attribute is not indicated. A fence is added whenever you use \left and \right; these commands must be followed by a delimiter (the list of delimiters is currently built-in). An <mrow> has been inserted between the positions marked l and r on the symbolic representation: On Figure 9 you see a variant of the formula (where an exponent has been added to make brackets greater), and the same formula, where the <mrow>s have been removed.

$$\{x,\ x^4 > \alpha\beta\}\cup\left[x\sqrt[3]{4}y\right] \text{ vs } \left\{x,\ x^4 > \alpha\beta\right\}\cup\left[x\sqrt[3]{4}y\right]$$

**Fig. 9.** Effect of <mrow> on the size of stretchable operators

Some operators are of variable size; we have seen examples of sums and square root previously. Some symbols are vertically extensible (for instance braces and brackets) their height is the height of the current sub-formula (the <mrow> or <mfenced>). In the example above, the height of the brackets is the height of the formula $[xy]$. If you say \big(, Tralics silently ignores the prefix before the parenthesis; in the case of \big(a^2\bigg), it is clever enough to understand that the opening parenthesis matches the closing one and uses fences.

Consider the following input

```
$ \mathcal{AB1\alpha}\mathbf{AB1\alpha}\mathrm{AB1\alpha}$
\mathfontproperty\mathcal=1
\mathfontproperty\mathfrak=1
$ \mathcal{AB1\alpha}\mathfrak{AB1}\mathrm{A}$
```

The translation of the first math expression is

```
<mi>&#x1D49C;&#x212C;</mi><mn V='script'>1</mn><mi>&#x3B1;</mi>
<mi>&#x1D400;&#x1D401;</mi><mn V='bold'>1</mn><mi>&#x3B1;</mi>
<mi> AB </mi><mn>1</mn><mi>&#x3B1;</mi>
```

Here V stands for 'mathvariant'. The following points are to be noted. First, the operators are unaffected by font changes (there is no slanted plus, no bold minus). Currently, Greek characters have a fixed translation although Unicode provides upright or slanted, regular or bold, Greek letters. Tralics uses only *mathvariant* for digits, even though variants of the characters exist in Unicode.

In many cases, the rendering is correct (see Figure 8, for an example of bold italic characters). Translation of `\mathrm` is tricky. It should produce an upright (non-italic) version of the object. We assume that this is the default case for digits. We also assume that is the default for a `<mi>` object that contains more than one letter. The translation of the second formula is

```
<mi V='script'>AB</mi><mn V='script'>1</mn><mi>&#x3B1;</mi>
<mi V='fraktur'>AB</mi><mn V='fraktur'>1</mn>
<mi V='normal'>A</mi>
```

You can see how `\mathrm` handles the case of a single letter: we use the *mathvariant* attribute with the 'normal' value. The two magic lines between the formulas tell Tralics to use the *mathvariant* attribute, rather than characters that are outside plane one. This is not not needed for Firefox when the Stix fonts are present.

## 5 Building Formulas

It is possible to obtain the following formula with Tralics

```
<apply>
  <power/><apply><plus/><ci>a</ci><ci>b</ci></apply><cn>2</cn>
</apply>
```

by typing

```
\newcommand\Apply[2]{\mathbox{apply}{\mathbox{#1}{}#2}}
$\Apply{power}{\Apply{plus}{\mathci{a}\mathci{b}} \mathcn{2}}$
```

Neither Amaya nor Firefox render the "content" part of the MathML recommendations, so that we consider here only "presentation" elements.

Details about the algorithms used by Tralics can be found in the report [3]. The first pass converts a list of characters into a nested list of XML elements, together with some commands. A recursive algorithm is used to combine the pieces, and evaluate the commands. The formula of the previous section may be represented as "liosiC(nn)oiiro(ii)", that contain a single command to be evaluated, the hat sign.

Each expression has a level (1 for the main expression, 2 for indices, 3 for subindices, etc), that can be changed in TEX via the use of commands like `\scriptstyle` (see Figure 6 for an example). The effective level is computed and `<mstyle>` elements may be inserted, while `\mathchoice` commands are reduced. Expressions of the form

```
$a\over {b \over c}$
$a^{^b}_{c_d}$
```

are considered. On the first line, there is a basic TEX expression, discouraged by amsmath: one should use a prefix version, namely `\frac` instead. The expression on the second line is considered so good practice that there is no prefix equivalent. If there was one, it would look like:

```
\supsub{a} {\sup{}b} {\sub c d}
```

In fact, Tralics does this conversion. To each kernel, for instance a, one can associate at most one subscript and at most one superscript (Note that Tralics has currently no support for tensors, i.e., objects that can have multiple scripts, on the left or right). It is legal to start an expression with an underscore or hat but not to finish it. The translation is

```
<msubsup>
   <mi>a</mi>
   <msub><mi>c</mi> <mi>d</mi> </msub>
   <msup><mrow/> <mi>b</mi> </msup>
</msubsup>
```

The kernel can be an operator like a sum. The placement of the index may depend on the mode (display or not); since Tralics knows the current mode, it used `<munder>` or `<msub>` (some MathML operators have a *movablelimits* attribute, and in such cases, the placement of the index could be wrong after all). We give here an example, where a, b and c are operators like $\sum$ in display style.

```
\def\X#1{\mathop #1\limits} $\X a ^{\X b ^2} _ {\X c _2}$
<munderover>
  <mi>a</mi>
  <munder><mi>c</mi> <mn>2</mn> </munder>
  <mover><mi>b</mi> <mn>2</mn> </mover>
</munderover>
```

There are several ways to put two objects one above the other. The `<munder>` and `<mover>` elements have a base object and a secondary object, which is generally smaller; `<munderover>` has two secondary objects; these elements can be produced by attaching scripts to a kernel in some special cases. The preferred way is to use `\underset` or `\overset`. In LaTeX you can also use `\xrightarrow`. Some commands like `\bar` or `\hat` produce accents (case where the element has an *accent* attribute). The effect of this attribute is often unclear. The recommendation says an accent is drawn closer to the base, and has a normal size. Both objects stretch horizontally. For instance, if you put text over an arrow, the width of the arrow is at least the width of the text; if you put a bar or a brace over a formula, it covers the whole formula.

The `<mfrac>` element puts two objects one above the other, they are generally separated by a fraction rule; it can be obtained by the `\frac` command, or variants thereof (the two objects are generally centered, of the same size; Tralics accepts `\hfill` command in math mode in order to change the alignment).

Arrays are handled by Tralics, as far as the result conforms to MathML. Horizontal alignment (rlc in the array preamble) is accepted, but column separators are ignored or rejected. This means that you cannot insert horizontal or vertical rules (there is a limited way of putting frame around a table

in MathML, but this is not yet implemented). Matrices and some math environments are converted by Tralics into tables.

It is possible to adjust horizontal or vertical spacing by adding explicit spaces (for instance `\mskip9mu` inserts 5pt of white space) or using phantoms.

## 6 Mixing Math and Text

In TEX, you can insert any box into a math formula (for instance an image, a reference to the bibliography, an external link, another math formula); this is not possible in MathML. An example of code refused by Tralics is

```
\xymatrix{A\ar[rd]{^f}&B\\C&D}
```

Our hope is that, in some future, there will be more interaction between XML standards, and a combination of SVG and MathML will render both the mathematical expressions and the arrows between them. A math expression like $\{x, \text{ such that } x > 0\}$ can be entered in TEX as

```
$\{x, \hbox { such that $x>0$} \} $
```

When Tralics sees non-math material inside a math formula, it provokes an error, unless this expression can be interpreted as combination of text and math. The algorithm is a bit tricky, and error messages may be confusing. We allow commands `\hbox`, `\text`, `\mbox`, and font changes inside them (they will be partially honored); we also allow math formulas inside text, and they will be rendered as math outside the text. The translation of the preceding example is

```
<mo>&lbrace;</mo><mi>x</mi><mo>,</mo>
<mspace width='4.pt'/>
<mtext>such</mtext><mspace width='4.pt'/><mtext>that</mtext>
<mspace width='4.pt'/>
<mrow><mi>x</mi><mo>&gt;</mo><mn>0</mn></mrow>
<mo>&rbrace;</mo>
```

Consider now the following formula : $\nearrow + \searrow + \swarrow = \triangle$ entered as

```
$\lower .97ex \hbox{$\rightarrow$} \mskip-24mu \nearrow + \nwarrow
\mskip-24mu \lower .97ex \hbox{$\leftarrow$} + \swarrow \mskip-2mu
\searrow = \hbox{$\diagup\mskip-1mu\diagdown$}
\lower.48ex\hbox{$\mskip-31mu\hbox to
  5.85mm{\strut\hrulefill\strut}$}$
```

The horizontal arrows are vertically shifted via the `\lower` command. This is currently impossible in MathML, as there is no equivalent of vertical row, vertical space, vertical adjustment. It happens that correct placement can be achieved in Firefox by using a subscript. On Figure 10, you can see that the same formula renders awfully in Amaya.

The right hand side of the equality contains a triangle, obtained by gluing two characters and using `\hrulefill` for the horizontal line. We have shown on the figure the rendering of the two characters by Firefox. As you can see, it is impossible to complete this into a triangle via an horizontal rule.

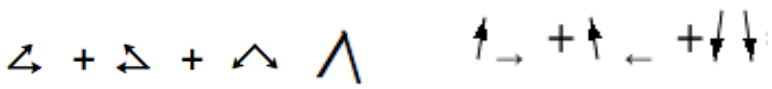

**Fig. 10.** A funny math formula, Firefox and Amaya

## 7 Conclusion

We have shown in this paper that Tralics is able to translate a great number of mathematical expressions into MathML, and that the result is often correctly rendered by browsers. There may be other uses of the translation (as feed to computer algebra systems, or for indexing purposes, data mining, etc.). We do not know how well the Tralics output behaves in these domains. In some cases, Tralics fails to correctly translate a document; it might be due to the use of big packages like *xypic*, or because people use non-math constructs into math formulas. Some useful packages should soon be adapted to Tralics, including packages that provide access to fonts, whether or not the characters are in the Unicode standard. We also plan to increase flexibility, in different areas of the software.

## References

1. Thierry Bouche. When CEDRAM meets Tralics. In: *Towards Digital Mathematics Library*, pages 153–165, Masaryk University, Brno, 2008. `http://dml.cz/dmlcz/702544`.
2. David Carlisle, Michel Goossens, and Sebastian Rahtz. De XML à PDF avec `xmltex` et PassiveTEX. In: *Cahiers Gutenberg*, number 35–36, pages 79–114, 2000.
3. José Grimm. Converting LaTeX to MathML: the Tralics algorithms. Research Report 6373, INRIA, 2007.
4. José Grimm. Producing MathML with Tralics. Rapport de Recherche 6181, Inria, 2007.
5. José Grimm. Convertir du LaTeX en HTML passant par XML: Deux exemples d'utilisation de Tralics. *Cahiers Gutenberg*, (51):25–55, 2009. October 2008, to appear.
6. Heinrich Stamerjohanns, Deyan Ginev, Catalin David, Dimitar Misev, Vladimir Zamdzhiev, and Michael Kohlhase. MathML-aware article conversion from LaTeX. In: *Towards a Digital Mathematics Library*, pages 109–120, Masaryk University, Brno, 2009. `http://dml.cz/dmlcz/702561`.
7. The Unicode Consortium. *The Unicode Standard, version 4.0*. Addison Wesley, 2003.