Interactive LaTeX to MathML/HTML translation

Brian Johnsen
brianj@mip.sdu.dk

Maersk Mc-Kinney Moller Institute for Production Technology
University of Southern Denmark, Odense

Supervisor : John Perram
Co-supervisor : Morten Andersen

January 2005

Contents

Chapter 1

Synopsis

This chapter summarizes this bachelor thesis.

The incitement for this bachelor thesis was to develop an application that could serve as a link between an old way of distributing scientific knowledge, by writing documents to be passed around as printouts or articles, to the new way: *The Internet*. When provided the ability to place LaTeX documents directly on the internet without any fuss, it should uncover a hole new way of presenting scientific material.

The thesis can be broken down into four main parts: An introduction to the problem domain and a motivation for doing this at all. An investigation into the various translation engines on the "market". An explanation on the developed software, and the decisions regarding it. And finally a discussion and a conclusion of the development.

Chapter 2

Preface

This chapter provides an introduction and the motivation for this bachelor thesis.

2.1 Abstract

Previously when publishing mathematical documents on the web, it was necessary to encapsulate the mathematics as images in the documents to ensure proper rendering of the formulas, or perhaps even typing the formulas as "ASCII art". This procedure was necessary because of the very limited features of HTML markup to render mathematics. Inevitably this procedure makes reuse of the mathematical formulas for further manipulation almost impossible.

2.1.1 LaTeX

Since the birth of TeX, created by Donald E. Knuth of Stanford University around 1980, and the creation of the *LaTeX document preparation system* shortly thereafter, LaTeX has become the de-facto standard for exchanging scientific documents. As of today, almost any mathematical publication are written in LaTeX. The reason for this, is because LaTeX has an almost unique ability to render mathematical formulas and symbols in the same way they are written in hand. Unfortunately LaTeX documents does not render very well on the Internet, unless they are compiled into PDF or PS documents, which makes them rather "static".

2.1.2 MathML

Mathematical Markup Language, or MathML, is a W3C [1] recommended standard for Internet publishing of mathematics. The intension of MathML is to capture both the presentation and the content semantics of mathematical expressions, and thereby providing the user high-quality visual display, and at the same time preserving the content of the mathematics. Because MathML encodes the content of the mathematics it is also accepted as input in various mathematical computation programs, such as Mathematica or

Maple. Unfortunately MathML it is too verbose to author directly, and so needs authoring tools that can translate from a standard mathematics-authoring markup such as LaTeX.

Although some editors have been made, providing excellent WYSIWYG interfaces to writing MathML directly, it would nevertheless be helpful to be able to convert a LaTeX document into MathML, without having to rewrite the hole thing. The rationale is of course the existence of a massive amount of scientific documents already written i LaTeX.

2.2 Goal

The goal of this Bachelor thesis is to make an application that translates a LaTeX source document into a MathML/HTML document, thereby making it immediately applicable on a web site or in a mathematical computation program.

Investigation of the various LaTeX to MathML translators available will be carried out, and if possible, one of them will be used as the base translator engine.

Because of the vastness and non-strict definition of the LaTeX system, only a subset of a LaTeX document can be immediately translated. The solution tested in this project is to prompt the user with the various LaTeX commands that can not be translated, and providing the user the ability to change these specific commands, thereby giving the user full control over the outcome of the translation.

Chapter 3

Translation Engines

In this chapter the various translation engines is tested and discussed for usability.

3.1 ASCIIMathML

Translating ASCII math notation to Presentation MathML
The ASCIIMathML translator is developed and maintained by Peter Jipsen of Chapman University.
(http://www1.chapman.edu/~jipsen/mathml/asciimath.html)

3.1.1 Evaluation of ASCIIMathML

This translator consists of a Java script which is meant to be used as an "engine" within a HTML or XML page. It has no direct support for ordinary text, much less support for the normal LaTeX document construction. It does however support a large chunk of the normal LaTeX math commands.
This translator is a great tool, if one needs to have math displayed within a web page. Nevertheless, for this project we would require a separate tool for parsing everything aside form the pure mathematical expressions within the source document.

3.2 Tex4ht

Tex4ht is developed by Eitan Gurari of the Ohio State University.
(http://www.cse.ohio-state.edu/~gurari/TeX4ht/mn.html)

3.2.1 Evaluation of Tex4ht

This translation engine unfortunately only converts into presentational MathML, consequently this engine has nor been tested further.

3.3 Tex4Moz

Tex4Moz is developed by Paul Gartside, and is a part of the MathZilla project.
(http://pear.math.pitt.edu/mathzilla/discussion.html)

3.3.1 Evaluation of Tex4Moz

The Tex4Moz translator is an extension of the Tex4ht engine mentioned above in section 3.2. It seems to be very sensitive about the structure of the LaTeX source document, and as such only accepts semantically correct latex. This means that potential source documents for this translator would have to be revised extensively.
At authoring time Tex4Moz has not been updated since March 2001, which probably constitute this project as dead.

3.4 Hermes

A content oriented LaTeX to XML+MathML conversion tool
Hermes is developed and maintained by Romeo Anghelache from the Max Planck Institute for Gravitational Physics.

3.4.1 Evaluation of Hermes

Hermes is a LaTeX to MathML conversion tool that helps converting LaTeX documents to MathML. It requires the use of TeX macros to add semantic information in order to generate XHTML+MathML documents from the LaTeX source.

3.5 Choice

Based upon the preliminary investigations of the "engine candidates", I decided to proceed with the Hermes engine. This choice was based on various accounts. For one, it is an ongoing project, which causes updates on a regular basis. The latest update is from November 2004. Secondly, it works with a wide amount of LaTeX commands, and needs relatively little alterations in the source documents to provide satisfactory translation results.
Chapter 4 is devoted to further scrutiny of the Hermes translator.

Chapter 4

The Hermes Translator

4.1 Overview of Hermes

The Hermes translator works in a three stage rocket:

1. LaTeX needs to be called on the source file, with a specific input parameter inserted in the document.

2. The Hermes translator then translate the document into MathML+HTML.

3. Finally the result is "parsed" through an XSLT processor to optimize the rendering of the document.

4.2 Discussion on Hermes

Testing of this engine revealed some specific LaTeX commands that the translator was unable to handle or simply ignored. These commands and their effect on the translation are described in appendix A.

As it is obvious from table A.1 the one major drawback of Hermes in the present release, is that it has no support for the `array` command which means that it is not possible to have any matrices in the source document. Hopefully this will be added in a coming release of Hermes.

Chapter 5

Software

This chapter investigates the software: requirements, decisions and implementation.

5.1 Vision

The vision of this project, as stated in section 2.2, is to develop an application that handles translation from a LaTeX input to a MathML/HTML output. Furthermore, as the translation engines are not fully developed or only developed to handle a specific subset of the LaTeX commands, it is of paramount importance to prompt the user, if the file to be translated contains any of the translator specific invalid commands. Thus enabling the user to make the appropriate adjustments in the file before proceeding with the translation.

5.2 Application Requirements

The basic requirements to the application is as follows:

- The application should provide the user with the ability to translate a LaTeX file into MathML and HTML.
- The application should parse the file for invalid commands and present them to the user, if any.
- The application should provide the user a way to interactively edit the source document if it contains any invalid commands.
- If the file contains no invalid commands, the file should be translated and returned to the user.
- The application should naturally provide a way of retrieving the translated document again.

Additional requirements:

Availability: To accommodate the notion of E-Learning [2], the service should be provided as a web application. Thus providing the use through the internet without requiring any additional installation of software on the client side.

5.3 Decisions

This section elaborates on the various decisions made concerning the implementation.

Programming Language
The application is programmed in Java and JavaServer Pages (JSP). This decision was made on the account of Java being "platform independent", which will prove conveniently on account of the chosen translation engine, and it is furthermore this authors preferred programming language.

Translation Engine
As explained in section 3.5 the chosen engine is the Hermes translator. This engine, however, requires to run on the UNIX operating platform, which again favors the implementation to be web based and the use of Java.

UNIX Platform
Because of various issues regarding the use of a complete UNIX platform, it was decided to adopt the Cygwin UNIX emulator in the project. Cygwin provides full UNIX emulation on a Windows platform and was an evident choice due to the fact that the rest development was done in a Windows environment.

Application Server
The Apache Tomcat Applications server was chosen for that assignment. First of all it is a free open source product that is the official servlet container of JSP. It furthermore works seamlessly with the Eclipse IDE, which was used during the development.

5.4 Implementation

This section makes a short description of the role of each Java class and JSP page of the system.

5.4.1 upload.jsp

This page handles uploading the source document to the server. It does that by using an uploadbean[1]. This uploads the file through the use of a `MultipartFormDataRequest`-object.

5.4.2 upload2.jsp

This page calls my parser which scans through the source document for illegal and ignored commands and stores them in two separate files. It then displays the the illegal and ignored commands to the user, if any. This is done by reading the generated files.

5.4.3 tokenHandler.jsp

This page provides the ability to retrieve the line that contained an illegal command, and a selected number of its surrounding lines. It then displays those lines in a textarea making it possible to remove those illegal commands.

5.4.4 tokenReplaceHandler.jsp

This page displays the altered lines back to the user, thus providing an ability to regret the alterations. If they are approved, it stores them back in the uploaded file.

5.4.5 hermesHandler.jsp

This page calls the Hermes translator through a provided class method. It then makes the fully translated file available for download.

5.4.6 LatexFile.java

This class works as a Javabean, providing storage and methods for the JSP pages.

5.4.7 Parser.java

This class handles the parsing of the uploaded file for illegal and ignored commands, and stores them in two separate files. It furthermore handles insertion of the Hermes input parameter needed by Hermes to translate the document.

5.4.8 TokenLib.java

This class works as a library of the illegal and ignored commands.

[1] the uploadbean is provided by Http://www.javazoom.com

5.4.9 ExecuteShell.java

This class handles the calling of the Hermes translator. This is done through an invocation of the `bash`-shell, which runs a script that controls the running sequence of Hermes.

Chapter 6

Discussion and Conclusion

The time to reflection has come.

6.1 Conclusion

Several translation engines has been reviewed and tested for their usability in relation to this project. One, the Hermes translator, was chosen to be the base engine for the software system.

The developed software system has been tested with various LaTeX source documents, and is working as expected. The output from a translation is fully renderable with either Mozilla or Firefox with the necessary fonts installed, and with IE with MathPlayer[1] installed.

The appearance of the various pages has been considered a secondary concern. This decision was based on the fact that the main goal of the thesis was to provide a dependable translation and not a fancy appearance. However it could need some work on the style to make it more "user friendly".

6.2 Future Improvements

It is this authors intent to keep working on the system, extending it with boat a more appetizing appearance and providing some validation on the users of the system and the files uploaded.

Another major pending improvement on this project is the lack of "separation of concerns". In other words the separation of the model and the control from the view.

One obvious way of handling that would be to implement the Model-View-Control (MVC) design paradigm. This could be beautifully adopted through the Jakarta Struts

[1]MathPlayer can be found at: http://www.dessci.com/en/products/mathplayer/

framework[3], because this framework provides a native MVC structure.
It furthermore provides a powerfull way of coordinating between the layers of the MVC structure through the use of the `ApplicationResourses.proporties` and `struts-config.xml` files, the latter handling all coordination between the model and the view layers, thus enhancing the maintenance of the system and making it more flexible.

It would moreover provide a more flexible system for further enhancements and future maintenance.

6.3 Acknowledgement

The author would like to thank Morten Andersen for providing ideas how to approach the various obstacles that arose throughout the writing of this thesis.

Appendix A

Hermes illegal commands

Table A.1 (page 16) is a list of LaTeX commands that the Hermes translator is unable to parse. The table furthermore explains the effect each command has on the translation. *For explanation of the intended effect in LaTeX see "The LaTeX Companion"[4]*

The first block of the table is concerned with math related commands, the second with textual related commands.

The important thing to notice in this table, is the commands that causes the translation to crash.

Fonts supported by Hermes: `cmr`, `cmsl`, `cmt`, `cmti`, `cmtt`, `cmmi`, `cmmib`, `cmsy`, `cmbsy`, `cmb`, `cmbx`, `cmex`, `msam`, `msbm`, `cmssdc`, `cmss`, `eufm`, `cmcsc`, `wncyr`, `rsfs`. If the source document uses a font which is not in this list, Hermes will complain and die.

As it is apparent of table A.1, Hermes does not support the `array` command. This is `the` major drawback of Hermes, because this completely eliminates the possibility of using matrices in the source document.

Command	Effect
array	**crash**
big, Big, bigg, Bigg	ignored
bigodot	ignored
bigoplus	ignored
boldmath	ignored
boldsymbol	ignored
hookleftarrow	is not rendered right
Join	is not rendered right
left, right	ignored
lefteqn	ignored
Longleftarrow, Longrightarrow	is not rendered right
mathbf	effects entire displaymath-block
mathrm	ignored
mbox	**crash**
pageref	ignored, but blank space is inserted
phantom	ignored
quad, qquad	ignored
section	turns text color blue - TEST AGAIN
sqrt	ONLY square root, rest is not supported
underline	ignored (not supported yet)
displaystyle	ignored
textstyle	ignored
scriptstyle	ignored
scriptscriptstyle	ignored
flushleft, flushright	ignored
center	ignored
newline	ignored
newpage	ignored
tabular enviroment	**crash**
textsf	ignored
texttt	OK, but **crash** if *emph* is used within

Table A.1: Table of illegal LaTeX commands in Hermes

Appendix B

Source Code

The source code is available on the enclosed CD-ROM, in the directory: "src".

Appendix C

General Use and Installation

This appendix describes the general use and installation of the developed software.

C.1 General Use

1. Upload page:
   The user has the ability to upload a source document. This is done in three steps:

   (a) The user selects the desired file from his own computer.

   (b) The user then must press `[Upload]` to upload the file to the server.

   (c) The user then proceeds to the next page.

2. Token page:
   The user is presented with the illegal tokens, divided into the two categories. The user now has the ability to proceed and make the necessary changes in the uploaded document:

3. Handling Bad Tokens:
   The user can now select the line(s) that contain an illegal command, and select the number of surrounding lines.
   When he has selected the line he wants to edit, it is displayed in a textfield.
   The displayed lines can now be modified at will, and committed to the original uploaded document.

4. Translation page:
   When no more illegal commands exist in the document, the user can now translate the document to XML.

5. Download page:
   When translation of the document is finished it can be downloaded.

C.2 Installation

First of all the system requires Cygwin to be installed at: "c:\cygwin". It furthermore requires a running Tomcat application server[1]. The filesystem `latex2mml` provided on the enclosed CD in the directory "latex2mml", can then be dropped into the "webapps"-folder and run.

C.3 Running Example

The system can also be viewed on my personal system on:
Http://62.66.134.153:8080/latex2mml/

[1]The system has only been tested on the Tomcat 5.5.4 application server.

Bibliography

[1] World Wide Web Consortium (W3C). *MathML*. http://www.w3c.org/Math/, 2004.

[2] Morten Andersen. *Exercise Publisher - Evolving education*. Master's thesis, MIP, Odense, 2002.

[3] Apache Software Foundation. *Struts*. http://struts.apache.org/, 2004.

[4] Michael Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, MA, USA, 1993.