A WWW Learning Environment for Mathematics

Kostadin Antchev, Markku Luhtalahti, Jari Multisilta, Seppo Pohjolainen, Kari Suomela

Abstract

This paper discusses the components and implementation of a hypermedia-based educational system for the mathematical sciences. Existing (RTF, LaTeX) lecture notes are converted into HTML documents, and hypertext links are generated automatically into them. The hypertext courses contain hypertext, problems and examples with hints, and interactive exercises. A computer algebra system is used to generate and randomize exercises and to check the students' answers. Email, bulletin boards, and videoconferencing software are available on the system to enhance interaction between the teacher and the distance learners. ***Keywords:*** *Hypermedia, learning environment, mathematics, conversion, automatic link generation*

Introduction

In this paper the design and implementation of a hypermedia-based learning environment on the World Wide Web (WWW) for the mathematical sciences will be discussed. This paper is an outcome of a national scale project lead by the Hypermedia Laboratory at Tampere University of Technology (TUT), supported by the Academy of Finland and the Finnish Ministry of Education. One of the purposes of the project is to produce mathematical hyperbooks on the WWW, to be used nationwide at schools, institutes, and universities, where the Internet is available. As such, Internet provides an inexpensive medium to transfer the material to different sites, and with WWW browsers the courses can be studied in different environments (UNIX, PC, and Macintosh).

The pilot material consists of two courses: a basic mathematics course for students starting their studies at university level and a course on matrix algebra. The purpose of the first course is to revise school mathematics and to provide help and support for mathematical problems that students will be facing. The structure of both of the courses consists of text, mathematical dictionary or database of definitions, exercises, examples, and computer-aided interactive exercises. The mathematical concepts in the lecture notes are linked to the corresponding definitions in the mathematical dictionary or the definition database to provide online help for understanding them. Exercises, with two hint levels and examples, are given as a part of the lecture notes. The symbolic algebra package, Mathematica, is used for interactive exercises; it generates different numerical and symbolic exercises, and checks the answers given by students.

The structure of the hypermedia courses is based on the experiences that the authors have got from constructing a standalone hypermedia learning environment for Macintosh [7, 8]. Positive classroom experiences encouraged us to extend the basic design ideas for this WWW version.

Communication between distant learners and teachers will be supported by bulletin boards, email and videoconferencing software. The data transfer speeds at the university vary from modem speeds of 14,4 Kb/s up to 155 Mb/s using ATM-network.

The presented hypermedia courses will be used in an experimental way to support distance learning through the Internet. A special group for

distance learners will be formed in fall 1995 and experience on distance education with the proposed system will be collected and analysed.

Converting Lecture Notes into HTML documents

The WWW courses can be based on existing textbooks or lecture notes. In our department quite a lot of lecture notes have been written with Microsoft Word (RTF) or LaTeX. In both cases the raw text can easily be converted with existing public domain (PD) converters. The quality of the converted text is rather good, but major problems were encountered due to the specific features of the mathematical notation, such as subscripts, superscripts, Greek alphabet, and other mathematical notation not supported by HTML 2.0. In the future, when the browsers support HTML 3.0, it will be possible to use super- and subscripts in a much more elegant way. However, mathematical formulas have still to be converted from RTF or LaTeX to HTML 3.0. The quality of the final output depends on the WWW browser and the size of the selected font. Changing the font size in the browser does not scale the pictures, which is a bit annoying.

From LaTeX to HTML

There are several PD converters that convert LaTeX to HTML. Our choice was LaTeX2HTML by Nicos Drakos from the University of Leeds [4]. As pointed out by Drakos this conversion is inherently difficult. This difficulty stems mainly from the fact that LaTeX is programmable.

A mathematical dictionary was written with LaTeX, and it contains many postscript figures and mathematical formulas. Mathematical formulas were converted into GIF format by LaTeX2HTML. This results in quite a lot of small picture files for each page. Due to the difficulty mentioned above we have implemented some LaTeX macros in the dictionary as Perl scripts. These macros are used mainly for including pictures in HTML documents.

From RTF to HTML

An RTF file contains text, formatting commands, pictures, and formulas, and is used to transfer documents between different word processors or between Macintosh and PC. Many word processors can read and write RTF.

In the Macintosh environment RTF files can be converted to HTML documents using a PD converter called RTFtoHTML [5]. However, it does not convert super- and superscript or Greek letters to GIF pictures. This is why a preprocessor was programmed to read the RTF files, to search for super- and subscripts and to write them as RTF representations of pictures into the corresponding places in the RTF file.

RTFtoHTML converts pictures and formulas to Macintosh pictures (PICT). They have to be converted to GIF format using another converter. Most of the WWW browsers read GIF files that are binary compatible, so that GIF files can be created in Macintosh and then transformed to UNIX, where many WWW servers exist. Graphics files should be saved in transparent mode, so that they will be displayed well even if the background color in the WWW browser is not white.

In order to handle Greek letters the RTFtoHTML converter was customized by modifying certain configuration files. In our case the Greek letters are displayed as GIF pictures. Every time the converter encounters a Greek letter it places the corresponding HTML tag in the HTML file.

Creating Links Automatically to HTML Documents

Our WWW courses contain several hundred HTML files. Thus it was necessary to create hypertext links automatically. The material was divided into files so that a subchapter corresponds to a file. In addition, the definition of each mathematical concept was saved to a file. This file structure made it possible for us to implement a powerful tool for automatic linking, called Linktool.

Purpose and Usage

Linktool was written in C and it runs on a UNIX platform. The purpose of the program is to add links to HTML documents. The program is fairly easy to use. The user just has to write the hot words into the so-called link file.

The format of the link file is simple. A line in the link file contains just one hot word or phrase, which is the anchor of a link, and the corresponding target. The target of link a is a path to an HTML file or any URL. In order to handle phrases with several words, such as "Normal Matrix," they must be written into the link file before single hot words, such as "Matrix."

Technical Aspects

In the Finnish language there are many inflected forms of a word. Other European languages sharing this feature are for example Estonian, Hungarian, and Bulgarian. Because of inflected words, it is difficult to identify all the occurrences of the hot words in the text.

Linktool solves this problem in the following way. First it tries to find out the stem of the word. When the stem is found, the program adds nearly all the possible endings to the word. The program uses regular expressions as search terms, in special GNU regexp package and strsed function, written by Terry Jones. Both of these are public domain software available for example from [10]. The next example shows how the program adds some Finnish word endings using regexp engine. The regexp, given below, matches with the basic form of a word "matriisi" (matrix) and the inflected forms of "matriisit" (matrices), "matriisina" and "matriisin."

```
matrii[bcdfghjklmnpqrstvw][bcdfghjklmnpq
    rstvw]?([aeiouy]|&auml;
    |&ouml)([aeiouy]|&auml;|&ouml;)?t?(n
    a)?\(ta)
    ?(n)?(en)?(jen)?(in)?(ssa)?(lla)?(ll
    e)?/
```

Interactive Exercises

A computer based learning environment should provide feedback for the students. Interactive online exercises form such an activity.

Overview

In a typical interactive exercise the computer will pose a problem and ask the student to fill in an online form. In some cases the user may be asked to answer the problem. The answer will be assessed, and a message will be returned describing whether the answer is correct or what went wrong. In some cases the form may serve as an aid for solving the actual problem. By submitting such a form the students may get a plot and/or results of calculations intended to help them, as it is for "Conic Sections"—an example described in the next section.

Often we would like to randomize a part of the problem's data, so that the students can work out several problems of the same type, successively. They may explicitly ask for "new values" to be generated by pressing a designated button, or they may simply arrive several times at the same node. From the authors' perspective, the users' inputs and the browsing history are valuable sources of information that are needed to improve the quality of the hypermedia materials.

Using Mathematica on WWW

The existing client- and server-side standards on the WWW support the development of distributed interactive applications using standard HTTP clients and servers. Our interactive exercises are such applications. For mathematics the ability of the server side to perform nontrivial numerical computations and symbolic manipulations is essential. Therefore, the idea of incorporating an advanced computer algebra system as part of the server-side software is natural. Mathematicareg. is such a system with powerful numerical, symbolic, and graphical capabilities [13]. It was chosen, because of its excellent communication protocol, namely MathLink [14], which

allows other programs to use the whole power of Mathematica.

As front-ends for the exercises we use WWW browsers that support forms as defined in HTML 2. Certain text formatting capabilities are also necessary. In particular, to present mathematical formulas in the exercises we use a two dimensional form, made up of several preformatted lines of characters, which we surround by <PRE> and </PRE> tags (abbreviation from preformatted). In fact, this looks just like normal Mathematica syntax. Plots, which are created using Mathematica and then converted to GIF format, are best to present inlined in HTML documents along with explanatory text and other information.

Mathematica's programming language, a high level interpreted language with a vast amount of built-in functions and a sophisticated pattern matching mechanism, has been used to code the algorithms for generating the problems' data, processing of the forms' contents, and creating aesthetically pleasing plots. The availability of various Mathematica packages further simplifies many programming tasks, especially in mathematics. We have also written our own packages—one per interactive exercise—to encapsulate the code for each exercise and for ease of its maintenance.

Implementation of Exercise Server

The server side of interactive exercises (or more generally interactive documents) is implemented as an extension of a standard HTTP server. CGI is the standard for running such extension programs under HTTP servers [9]. Consequently, these programs are referred to as CGI-handlers (gateways and CGI-scripts are also in use). In UNIX, the HTTP server spawns the handler which is referred to by the URL of an incoming information request. The handler being invoked should return the appropriate HTML or other type of document, or it should generate such a document on the fly. Our CGI-handlers fall into the second category—an HTML document is generated and returned to the HTTP-server.

For the reasons outlined above, our CGI-handlers also use Mathematica by acting as clients of another server which incorporates it. It is worth mentioning that it is not desirable for each CGI-handler to launch a new copy of Mathematica whenever it is being invoked. First, it is largely time-consuming because of the initialization performed by Mathematica at startup—it may take up to 20 seconds on a well-loaded DEC Alpha station. Second, it may cause license violations if the number of Mathematica processes exceeds a certain limit. The solution was to implement a server which incorporates a single Mathematica process. This server is referred to as Exercise Server (ES) and, like the standard HTTP server, is running as a daemon process under UNIX. ES makes it possible for the CGI-handlers to redirect the incoming HTTP requests to Mathematica. It is Mathematica, appropriately programmed, which actually generates our interactive exercises and processes the forms' contents. To clarify this, an overall block diagram of the server side is given in Figure 1.

In most of the exercises there might be several requests which form a session. This entails that for each session certain state information has to be maintained and special care should be taken to support more than one session at a time. In this respect the HTTP protocol is stateless. Connections to the HTTP server are made only at the user's request, and between connections the HTTP server is not maintaining any information for or about individual users. Therefore there is no direct support, at the HTTP level, for interactive sessions which consist of several HTTP requests. It is the application's own responsibility to preserve such information. There is, however, a mechanism for accomplishing this in HTML 2. State information can be passed in special "hidden" fields to the clients which will resubmit it later along with the subsequent request [6]. In some applications the entire state information, that is, all the internal data structures which must persist, is flattened out and either written to a file or passed to the client as one or more hidden

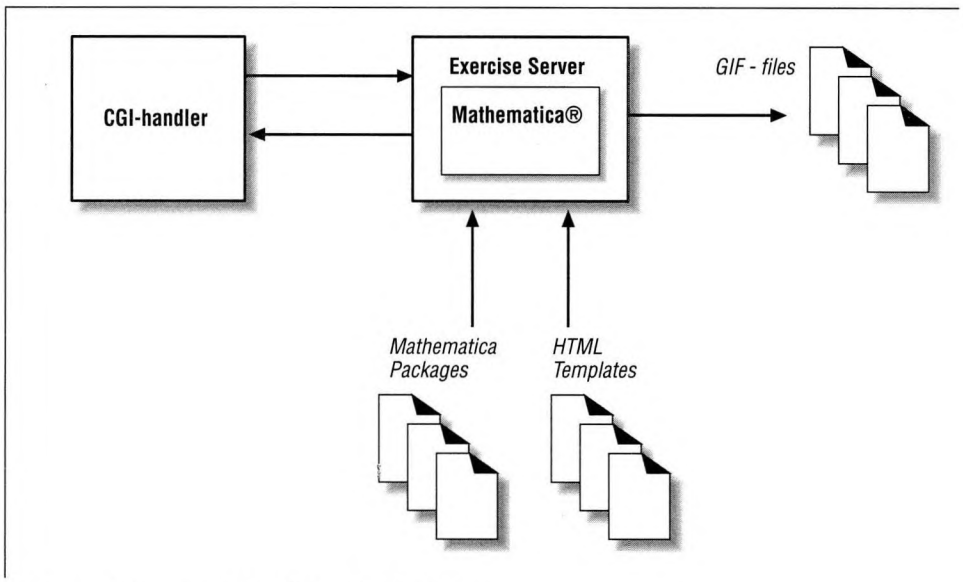

Figure 1: The server side of interactive exercises

fields [2]. However, in our case, the use of Exercise Server simplifies the task of maintaining the state information. It is no longer necessary to encode/decode data structures to/from flat representation at each HTTP request because it can be kept throughout the entire session in Exercise Server's main memory. Consequently, only an identifier of the session is necessary to be passed in a hidden field. A convenient built-in structure to keep the state information is the context tree [13]. Contexts in Mathematica facilitate distinguishing different uses of a particular name and, as such, are used by our interactive exercises to store the state information of concurrent sessions.

Upon the client's request, the HTTP server spawns the corresponding CGI-handler and passes the encoded form's content to it (assuming the request is using the POST method; see Figure 1). The latter decodes the form's content and connects to ES. The communication between the CGI-handlers and ES is based on Berkeley Sockets API [11] and allows ES to be run either on the same computer, where the HTTP server and CGI-handlers are running, or on a remote one over a TCP/IP network. An equivalent of the original request is forwarded first to ES and then via MathLink to Mathematica. It is either of the following commands:

```
DoOpenDocument["docname"]

DoSubmitForm["formname","con-
text",{"input1", ...,"inputn"}] ,
```

which directly comply with Mathematica's syntax. The first command explicitly refers to the exercise to be opened. It is an equivalent of the initial GET request. In response, inside Mathematica's environment, a session is initiated and a new context is devoted to it. Inside this context, the state of the session can be conveniently stored as values of symbols, safely isolated from other concurrent sessions. The name of the context is returned in a hidden field, which is part of each form which the generated document may contain. The second command corresponds to a POST request and always refers to a particular session by the name of its context. Formname is the name of the form being submitted. Input1 to inputn are the information entered by the user in

the form. There is no explicit command used to terminate a session. Instead, periodically ES examines all opened sessions and closes (deleting their context) those which haven't been active during a certain time interval.

ES replies to any request by generating an HTML-document. It is based on an HTML-template which holds the static parts of the document—those which do not depend on the interaction between the user and the server. The HTML-templates are ordinary HTML files which are normally created with an HTML-editor or any text editor. The places where interactive elements (formulas, plots, text responses, etc.) should appear are marked with designated strings. These strings are then replaced by ES with relevant HTML-elements in order to produce the final HTML-documents, which are sent to the clients. All the plots created at runtime are converted to GIF files using UNIX utility programs and their URLs are embedded in the documents. These GIF-files are best kept in the UNIX system's temporary directory so that they are automatically destroyed on a regular basis.

Example on Conic Sections

"Conic Sections" is an interactive exercise in the course of basic mathematics. It consists of several HTML pages which can be viewed with standard WWW browsers. The example seen in Figure 2 can be studied on our WWW server [1]. The example runs as follows.

Page 1. A second order equation describing a quadratic curve in the xy-plane is generated by Mathematica, and the student is asked to determine the type of the curve.

Page 2. The second page is displayed after the question on Page 1 has been answered correctly. The new problem, related to the same curve, is to bring the curve into its canonical form using coordinate transformations. The computations are carried out by Mathematica. Students are asked to enter the parameters of the coordinate transformations. To help them identify these parameters, the curve is plotted.

Page 3. The curve is presented in the new coordinate system both as an algebraic equation and as a plot. Students can see the effect of the coordinate transforms and compare their plots with the correct one. They may return to Page 2 to correct their transforms until the desired form is achieved.

Interaction Between Students and Teachers

Although the designed courses are suitable for distance learning and self-study, interaction between students and teachers is still desirable. The distant students should enroll in the courses, and they should know the order and the timetable in which the courses are being taught. They might have problems with the lectures and the exercises and would like to ask for advise or help. The final examination needs to be organized, and the results should be sent to the students. The designed hypermedia environment supports student-teacher interaction on several levels. For each hypermedia course a bulletin board will be set up. The bulletin board is in fact a Usenet newsgroup, and it is visible to every student. In the front page of the WWW course they find a button to open the bulletin board. The students may also put their messages on this board, if they like.

Students may also contact the teacher in a more personal way. At the designated places on the hypernotes they find buttons to send email to the teacher. This feature will be used to give the students personal assistance, to receive their solutions to given problems, and to give them personal feedback. The possibility of attaching documents with email is especially useful, since it allows the sender to point out details on the WWW material.

In addition to email connections and bulletin boards the students and their teacher can also

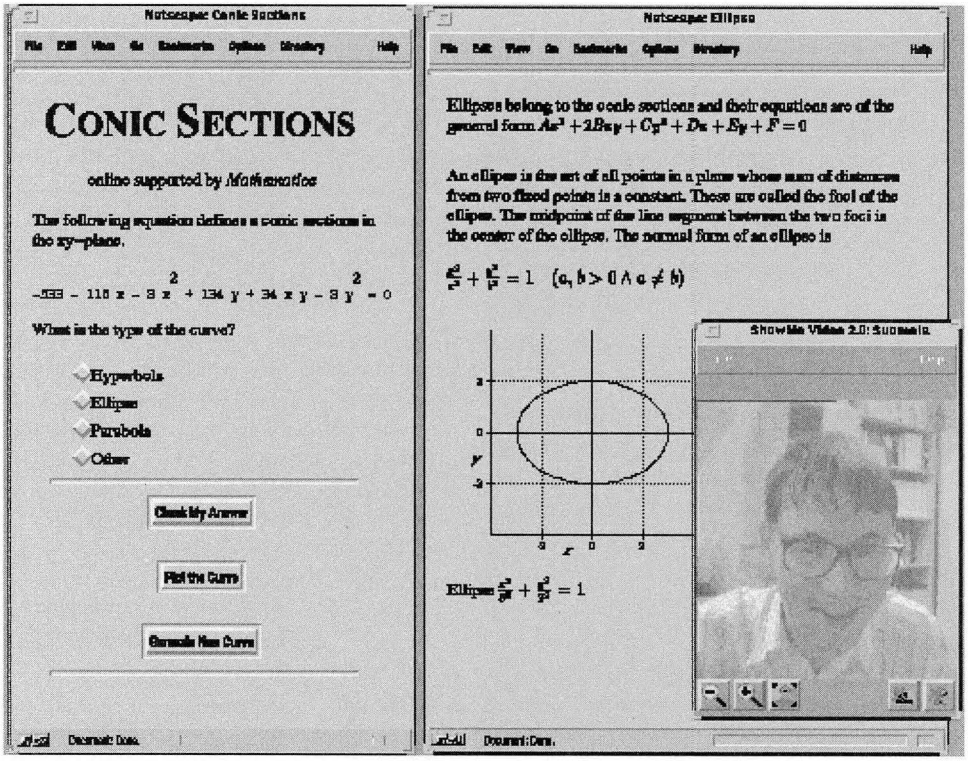

Figure 2: Learning environment with interactive exercise and video connection to the teacher

use videoconferencing software for their meetings and discussions. With ShowMe on Sun [12] and CU-SeeMe [3] on Macintosh and PC, it is possible to establish real-time videolinks. Generally, it is not possible to open a videolink from HTML document to a specific address automatically. Instead, the videolinks have to be opened manually by launching the videoconferencing software and entering the information needed to establish a connection to the teacher. However, the developers of CU-SeeMe videoconferencing software are planning to create a standard (i.e., URL specification) to open a videolink using HTML.

ShowMe includes audio, shared applications, video and whiteboard applications. It uses TCP/IP-protocol and supports Ethernet and ATM. With the Ethernet maximal speed of 10 Mbit/s picture refresh rate may be low and voice may be cropped. The available ATM-connections allow speeds of up to 155 Mb/s, so that the quality of video and audio is better.

Audio provides 8-bit 8 KHz at 64 Kbps. Audio traffic can be multi- or one-way. Shared Application works with X11-compliant applications. A user can start applications that the other users can see and use. Applications run in the local computer and only send display to other participants screens. Video lets the users see all the participants and take images from participants' video-windows. The quality of the video is not yet sufficiently good to show teaching with, e.g., chalk and blackboard. Text and pictures can also be shown using special shared application called Whiteboard. Participants can see the Whiteboard and write or draw messages on it.

Conclusion

In this paper a hypermedia-based learning environment which supports self-studying and distance learning on the WWW has been presented. Apart from the text and graphics conversions, the main components of the environment are automatically linked hypertext, a database of definitions, exercises and examples with hints, and interactive exercises where an online computer algebra system is used to generate and randomize the problems and to check the answers. Interaction between the teacher and the distant learners will be supported by bulletin boards, email, and finally with videoconferencing software. An experimental group of distance learners will be set up this autumn to test the proposed learning environment both from the technical and the educational point of view. ■

References

1. Antchev K., Luhtalahti M., Suomela K., Pohjolainen S., Conic Sections, Available at *http://matwww.ee.tut.fi/cgi-bin/ConicSections*
2. Paul Burchard (1995), W3Kit 2.2, An Object-Oriented Toolkit for Interactive WWW Applications, The Geometry Center, University of Minnesota, Online document available at *http://www.geom.umn.edu/docs/W3Kit/W3Kit.html*
3. The CU-SeeMe Project (1995), Available at *http://CU-SeeMe.cornell.edu/*
4. Drakos N. From Text to Hypertext: A Post-Hoc Rationalisation of LaTeX2HTML. *First International Conference of the World Wide Web*, 1994 at CERN, Geneva. Available at *http://cbl.leeds.ac.uk/nikos/doc/www94/www94.html*
5. Hector Chris (1995) RTFtoHTML 2.7.5 Converter for Macintosh, Available at *ftp://ftp.ncsa.uiuc.edu/Mosaic/Mac/Related/rtf-to-html-converter-275.hqx*
6. Internet Engineering Task Force (1995), HTTP: A protocol for networked information, Internet draft available at *http://www.w3.org/hypertext/WWW/Protocols/HTTP/HTTP2.html*
7. Multisilta J., and Pohjolainen S. Using Hypermedia in Teaching Linear Algebra. In John G. Lewis (ed.): *Proceedings of the Fifth SIAM Conference on Applied Linear Algebra*, Philadelphia 1994.
8. Multisilta J., and Pohjolainen S. Implementation of Authoring Tools for Hypermedia Based Learning Environments in Mathematics. *Proceedings of CALISCE '94*, 31.8-2.9. Paris, France, 1994. Available at *http://matwww.ee.tut.fi/Docs/paris/paris.html*
9. NCSA httpd Development Team, The Common Gateway Interface, Online document available at *http://hooboo.ncsa.uiuc.edu/cgi/overview.html*
10. Source Code Cdrom (1994), Walnut Creek CDROM, 1547 Palos Verdes Mall, Suite 260, Walnut Creek, CA 94596 USA, email: *info@cdrom.com*
11. Richard, Stevens W., UNIX Network Programming, Prentice Hall, Englewood, New Jersey, 1990.
12. Sun Microsystems (1995) ShowMe Product Overview. Available at *http://www.sun.com/cgi-bin/show?products-n-solutions/sw/ShowMe/index.html*
13. Wolfram S., Mathematica(TM) A System for Doing Mathematics by Computer, Addison-Wesley, Reading, MA, 1988.
14. Wolfram Research, MathLink Reference Guide, Technical Report, Wolfram Research Inc., 1992. Available at *http://www.wri.com/mathsource/*

About the Authors

Antchev Kostadin
Researcher, M.Sc.
kostadin@butler.cc.tut.fi

Luhtalahti Markku
Researcher, M.Sc.
luhtalah@butler.cc.tut.fi

Multisilta Jari
Researcher, M.Sc.
multisil@butler.cc.tut.fi

Pohjolainen Seppo
Ass. Prof.
pohjola1@butler.cc.tut.fi

Suomela Kari
Researcher, M.Sc.
suomela@harppu.ee.tut.fi

Tampere University of Technology (TUT)
Hypermedia Laboratory
PO BOX 692, FIN-33101, Tampere, FINLAND
http://matwww.ee.tut.fi/