MacQTEX: SELF-TESTING QUIZZES, USING PDF

ROSS MOORE AND FRANCES GRIFFIN

Abstract. The *MacQTEX* quiz system uses JavaScript [1, 8] embedded within PDF format [4] documents to allow students to do multiple-choice style quizzes. The internet may be used to supply the quiz document, and to record results. But even when not connected, there is immediate feedback as to how many questions were answered correctly and what are the correct answers, as well as providing worked solutions indicating how the correct answers could be deduced.

The highest quality of typesetting is employed in the quizzes by using the TEX typesetting software [7], via the pdf-TEX variant [6], to control the generation of the PDF documents [4]. Other software, such as Perl [10] and *Mathematica* [11], can be used to control the production of unique instances of a particular quiz so that each student gets slightly different questions to answer.

PDF Quizzes

At Macquarie University the Mathematics Department has been developing[1] a web-based system for producing quizzes which allow students to test their knowledge of mathematical ideas required in the courses that we teach. Currently these quizzes are used mainly at the most elementary level, for revision of the basic skills which the students should have acquired from mathematics courses at high school.

The current version of this quiz facility provides students with a multiple-choice answer quiz, of typically 10–12 questions, as a PDF document [4] downloaded from a web-site (figure 1). This document is an interactive form, controlled using embedded JavaScript [1, 8], which allows a student to read and work with the document, using the Acrobat Reader plug-in [2] to his/her favourite web-browser (figure 2). After starting with the "Begin Quiz" button (figure 3), answers may be selected and changed answers until the quiz is completed. Upon pressing the "End Quiz" button (figure 4), results of the student's attempts are submitted to a server for recording, provided that the network connection is still available. The embedded JavaScript then provides a means for the student to see which were the correct answers for each question (figure 5). Worked solutions, which were hitherto hidden, become available for reading and/or printing. A high quality presentation, both on-screen and when printed, is achieved by using the most up-to-date method of generating PDF documents [4] using the TEX typesetting software [7, 6]. This is especially important for the presentation of mathematics. Figures 3 through 7 show a sample of quiz questions and worked solutions.

Training not Testing

The aim of the quizzes is not so much for assessment as for self-testing and practice of material covered previously in lectures or back at school. At Macquarie, this is very valuable as our students come to University with a wide range of abilities and mathematical backgrounds. For many students it has been several years since they last studied any mathematics, but perhaps only a refresher is needed. For others, their main area of study is not mathematics at all, so there may be holes in their mathematical knowledge which need to be identified. We do not have sufficient staff to bring every student up to the level that would enable them to most easily cope with new material in the courses which they are about to undertake. The quizzes can be used by students to identify for themselves where they are weak and may need to seek the extra help that can be provided.

[1]This project has received funding via a 'Targeted Flagship Grant' from the Center for Flexible Learning, Macquarie University, and the Division of Information and Communication Sciences, Macquarie University as well as an equipment grant from Apple Computer, Australia Pty Ltd, via the Apple Universities Consortium.

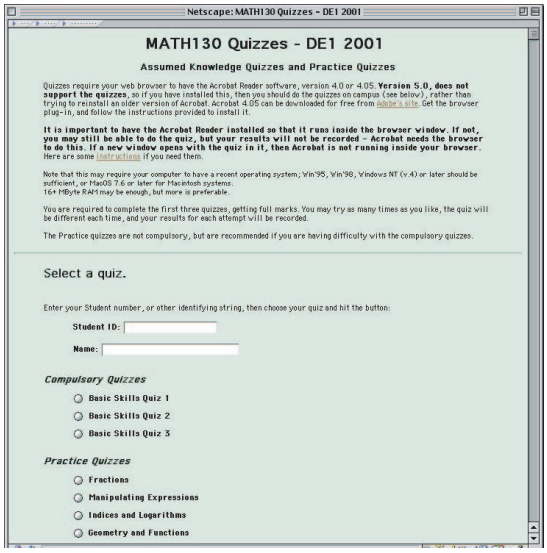

FIGURE 1. Quiz-site (at http://www.maths.mq.edu.au/~chrism/math130/MATH130quizzes.html) from which students can download the compulsory quiz documents. Username/password are required for recording accesses and results. Also from this site they may download practice quizzes, devoted to a particular mathematical concept. Guest access is also allowed for all quizzes.

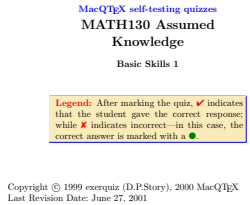

FIGURE 2. An opening page to a typical quiz.

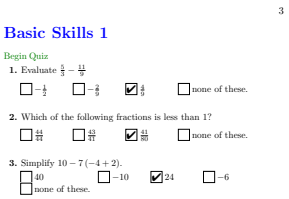

FIGURE 3. First page of questions, with "Begin Quiz" button and user-selections.

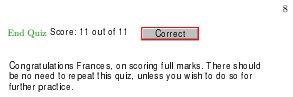

FIGURE 4. Last page of questions, after having selected the "End Quiz" button, showing the total score, and confirmation message.

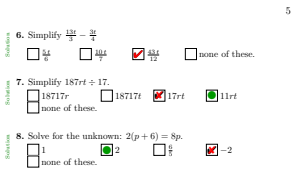

FIGURE 5. Embedded JAVASCRIPT [1, 8] is used to show the correct answer, when the student has made an incorrect choice. Also visible are buttons, previously hidden, which link to worked solutions.

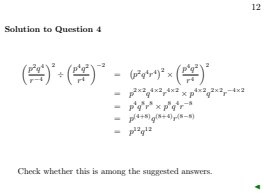

FIGURE 6. Worked solutions use properly typeset mathematics, as do the questions themselves. This one makes substantial use of mathematical symbols and equation alignments.

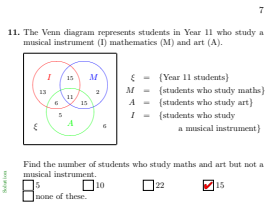

FIGURE 7. Elegant mathematical diagrams can also be used, both in the quiz questions and worked solutions.

Each Quiz is Different

An element of randomness has been incorporated into the generation of the questions for the quiz documents. Thus a student may attempt the 'same' quiz many times; but each time it will be slightly different. Since we teach mathematics, it is not hard to have some numerical aspect of the quiz being different for each instance. On the system that we currently employ for the elementary-level mathematics course, this is achieved using the *Mathematica*[2] software [11] to generate the exact contents of each question and its worked solution. Randomness is applied also to the specific choices presented as possible answers to a particular question, and to the order in which the choices are presented. Thus a student cannot 'cheat' by presuming, for example, that the correct answer to question 5 will always be 'b'. Indeed students are encouraged to repeat the same quiz as many times as is necessary to achieve a score of 100% correct.

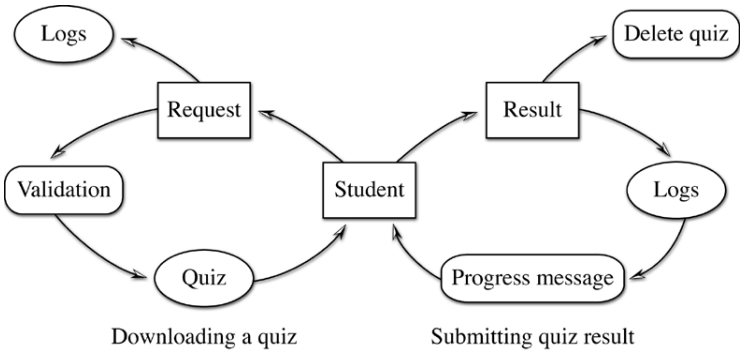

Figure 8. When a student requests a quiz, the student's identity is first validated; if authorized, the quiz is sent to the student's browser. A record is kept of all request details. After completing the quiz, submitted results are recorded in the student's log and in the overall quiz log. A personalized message concerning the student's progress is returned as FDF data [3]; this appears in a form field at the end of the PDF quiz document.

Logging Access and Answers

Part of the site administration is to keep a record of all attempts to download quiz documents from the site, and the return of results from completed quizzes. As well as this, records are kept of each student's access, which instance of a randomised quiz they received, what were the correct answers for that instance, and what results were obtained by the student.

The flow-chart in figure 8 indicates the interaction that a student has with the delivery and recording aspect of the overall system. Part of the "Validation" step is to setup the personal log-file for the student, or to append the new access information to an existing log-file. This same log-file is used to record the results obtain in each quiz instance accessed by this student. The information is available to the student, as shown in figure 9. Such pages can be reached from the initial quiz-access page (figure 1).

Evaluating Student Progress

The student log-file pages are also accessible from the staff interface page (figures 10 and 11); in fact all quiz logs can be viewed from here, including the raw log for any student and HTML formatted data extracted from all the logs (figures 12 and 13) for a particular course. This includes information such as the numbers of students who have attempted and/or completed the quizzes, the average number of attempts needed to complete each quiz, the maximum number of attempts taken, and the numbers of students who have downloaded each quiz, even though they have not been able to return any results (e.g., due to incorrect set-up of Acrobat Reader [2]).

[2] *Mathematica* is a trademark of Wolfram Research Limited.[11]

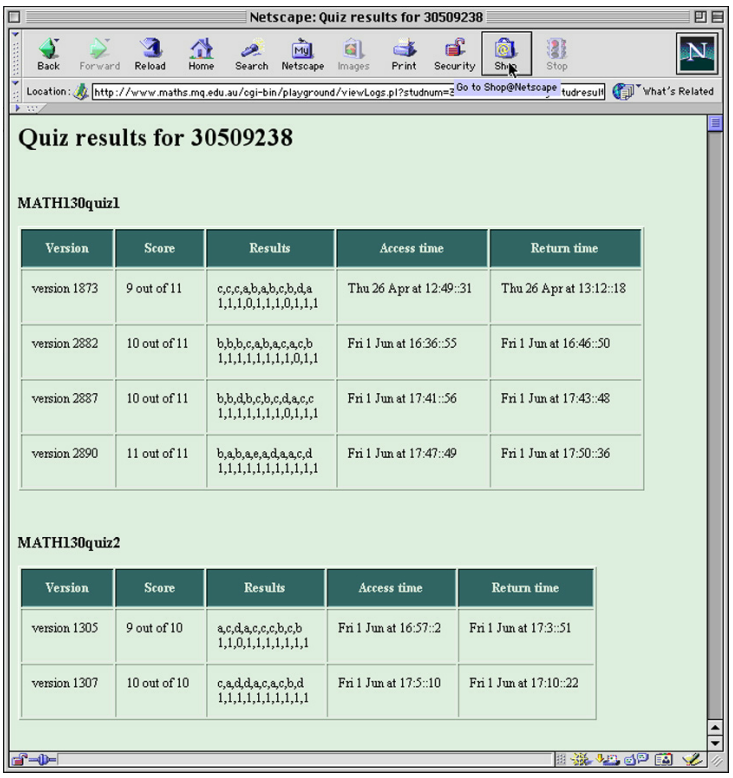

FIGURE 9. Students can view the contents of the file which logs their own attempts at the quizzes and records their results. Such pages are produced "on-the-fly" using a CGI interface, based upon a student's ID code. Instructors can also produce such pages, to easily see where a student is having difficulties.

Thus an instructor may assess the levels of usage of the quizzes, how difficult the students are finding them, and how the class is progressing in general. The ability to view the raw logs makes it possible to diagnose technical problems that students have, such as using browsers which don't support Acrobat forms [1, 3], accessing quizzes on campus from computers known not to be configured appropriately, and even tracing crashing problems on a student's computer to the day the browser software was upgraded!

DESIGNING A QUIZ

A bank of questions is available for constructing new quizzes. These cover 12 areas of basic mathematics, with up to 6 specific types of question in each area. Currently a new quiz is designed by choosing options in a form at a web-page (figure 15) accessible only with a password. This is only the first step in the process of creating quiz instances to be available for student use. The flow-chart in figure 14 indicates the main steps in the whole process.

After the choice of questions has been submitted, along with a directory name for the new quiz, a sample quiz document is generated and returned to the instructor for inspection. Editing of the LaTeX sources for the quiz is allowed (figure 16), as is adjusting the selection of quiz questions. When completely satisfied with the format and layout of the quiz, 50 randomised instances of the PDF quiz document are generated. Student log-files are set up. Now the quiz is ready for general access.

Each student access delivers the next in the sequence of 50 prepared instances. When the store of quizzes runs low, a script is triggered to automatically generate another 50 instances. (This number can be changed as desired.) Each instance of the quiz is unique, with the random aspects to the questions being created by *Mathematica* [11]. An interesting problem exists here, in that it is easy to program correct answers; but the incorrect answers must be believable also. By studying common mistakes that students make, it is possible to program *Mathematica* to make these same mistakes, thus providing some very tempting wrong answers!

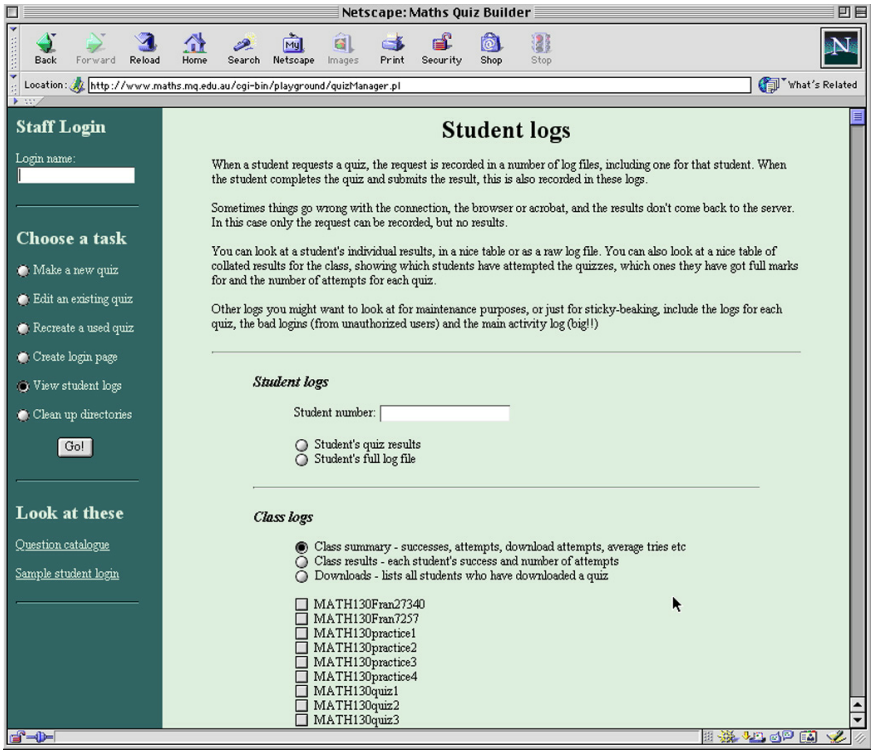

FIGURE 10. This page shows the access page for staff and instructors. It allows access to all of the log-files for the available quizzes, and for individual students. Information can be viewed in the raw text form, formatted as HTML tables, or some statistics can be collected from the logs. The page can be quite long; figure 11 gives the continuation after scrolling.

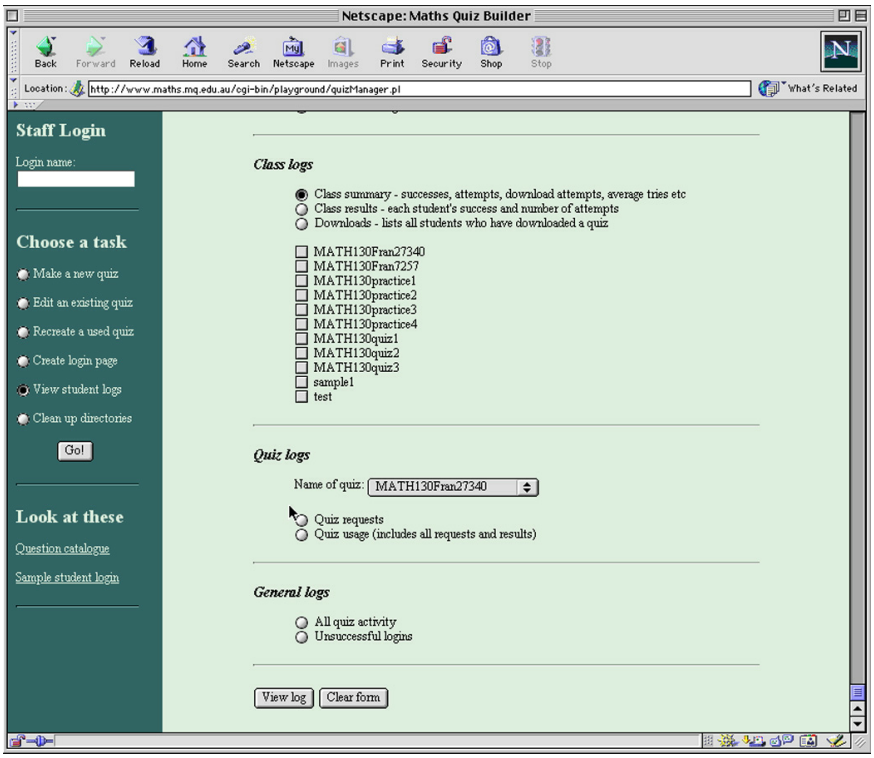

FIGURE 11. Continuation of the "staff-access" page, as described in figure 10.

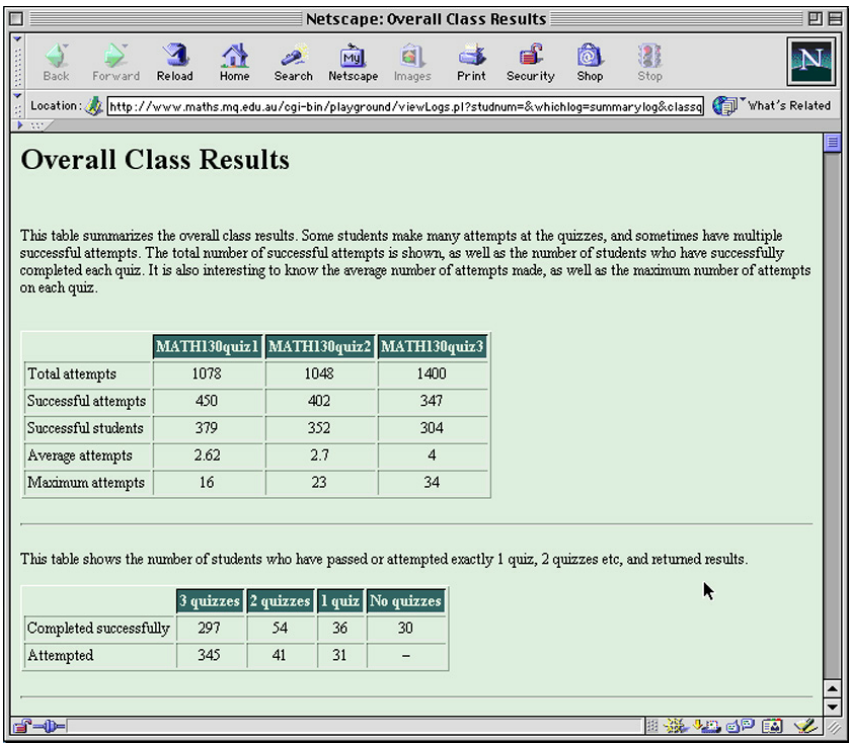

FIGURE 12. Summary statistics of the results of the quizzes for all students taking a particular course.

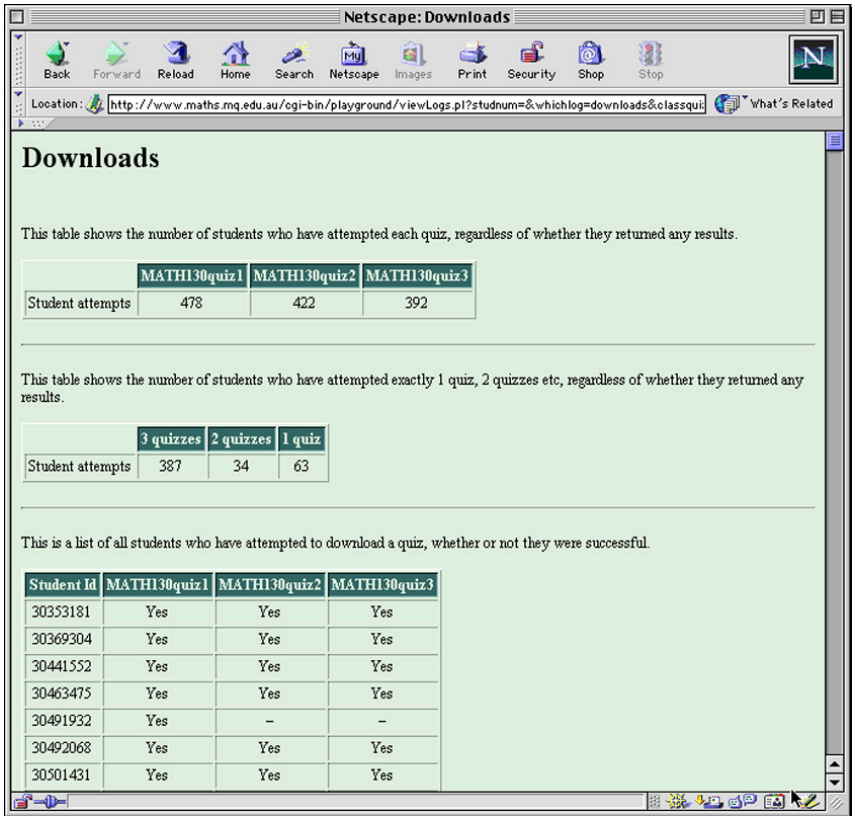

FIGURE 13. This page gives, for a particular course and set of quizzes, information concerning which students have attempted which quizzes.

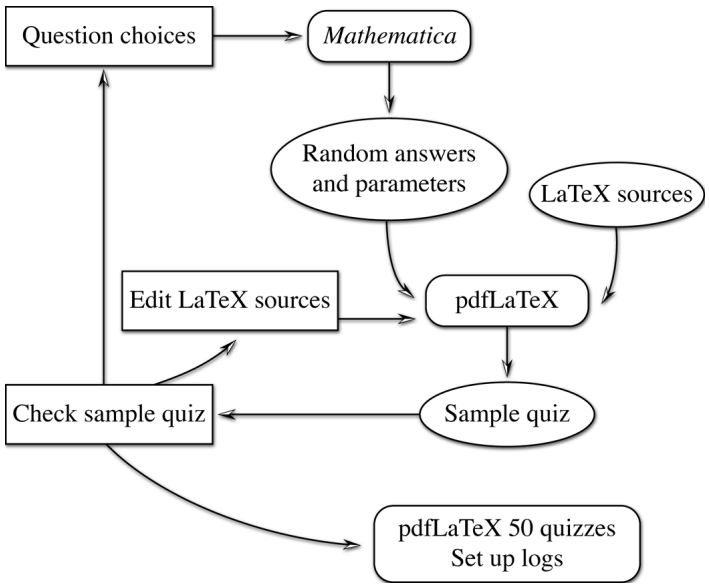

FIGURE 14. The questions for a new quiz are chosen using an HTML form. When randomisation is required, these selections are used to create input for the *Mathematica* program [11], which outputs parameters specific to each quiz instance, both for the answer options to each question, and for the worked solutions. These are combined with existing LaTeX sources [7] which control the overall layout, and processed with pdf-LaTeX [6] to produce a sample quiz in PDF format [4]. This is checked by the user, who has the option of making detailed edits to any of the LaTeX source files, changing selections, or rejecting the whole quiz and making completely different selections. When the sample PDF is finally deemed to be acceptable, the first 50 quizzes are generated, to be available for student access.

Further development of the system will be to create an easier interface for all of the stages of generating and checking new quizzes. For MacOS X, this will be an application[3] that can be run on the same server on which the quizzes are built. This should be much faster than using the existing web-based interface.

OTHER FEATURES

To facilitate managing the quiz system there are several options available. These include the following.

- An old quiz instance can be recreated from its numerical identifier.
  This is important so that an instructor may recover the exact questions that confronted a student who is seeking further help. It is also useful in case a student reports a problem with a particular instance of a quiz, indicating some sort of programming error (e.g., duplicated alternative answers, or a correct answer being declared incorrect, or *vice versa*).
- Existing quizzes may be edited in a number of ways. A quiz can be duplicated or edited in-place, and alterations made to the title, topic or question choices. In addition, certain text-only files may be downloaded for hand-editing to change aspects of wording or layout, then returned to the server. After a quiz has been edited, the logs and counters can be reset, and existing logs archived. This is desirable when the quiz is to be reused with a new class.

[3]The authors and development team wish to thank the AUDF for providing an Apple server and another machine for the development and testing of this application interface.

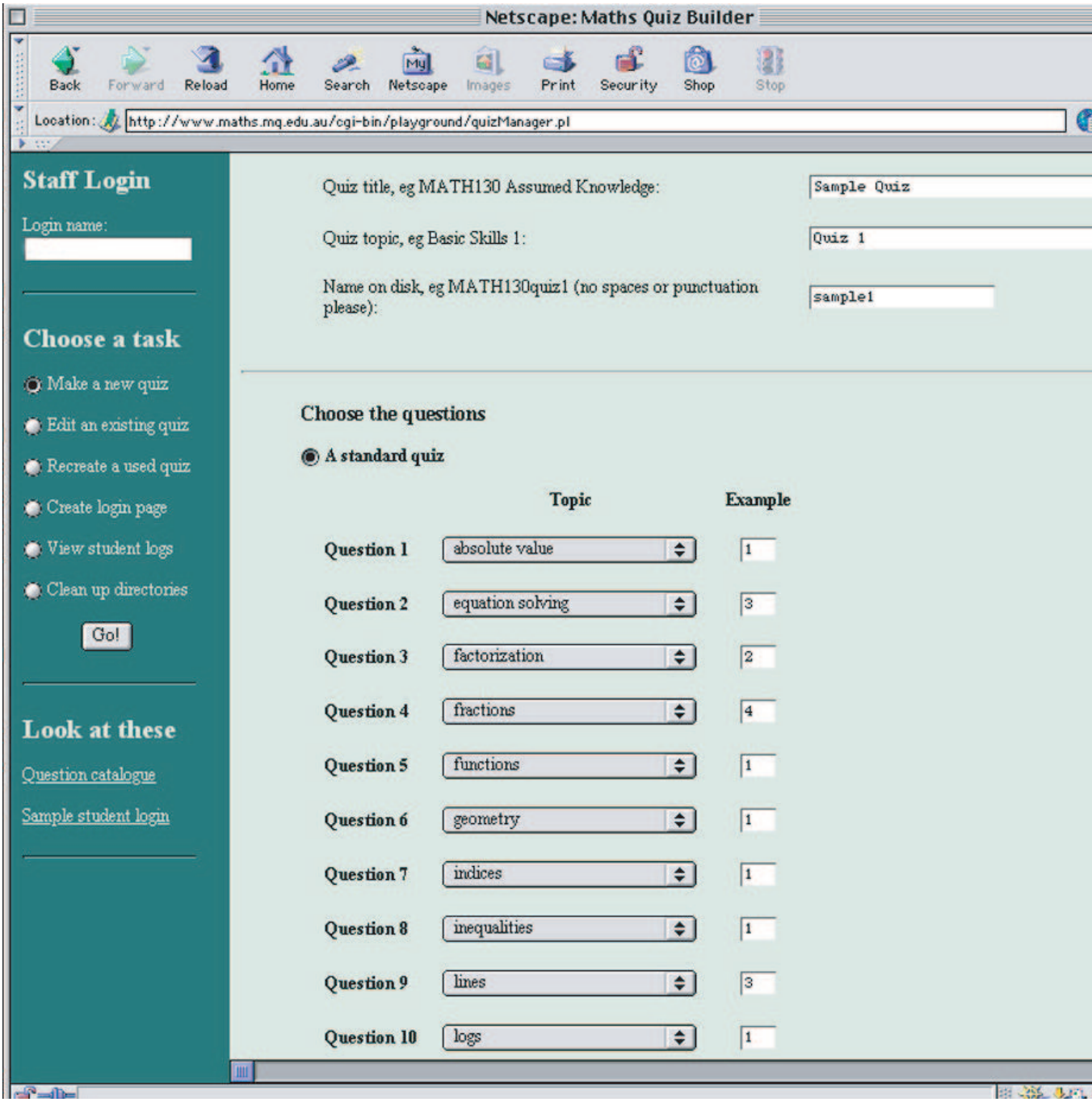

FIGURE 15. The quiz-design page, has pop-ups for basic topics in which questions are available, and a field to choose the specific question in an topic.

- House-keeping activities, such as removing quizzes that are no longer needed, can be done through the web interface. It is also possible to reset the quiz logs, archive these logs, or delete old archives.
- An HTML student-login page can be designed automatically. This includes the form fields and JAVASCRIPT [1, 8] necessary to reduce user error (e.g., omitting the login name, forgetting to choose a quiz, checking for the browser plug-in, etc.). The page can be downloaded and edited further to include information specific to the course for which the quiz is intended. It is not necessary to store the login page with the quizzes themselves; for example, it can be kept with other online materials for the course.

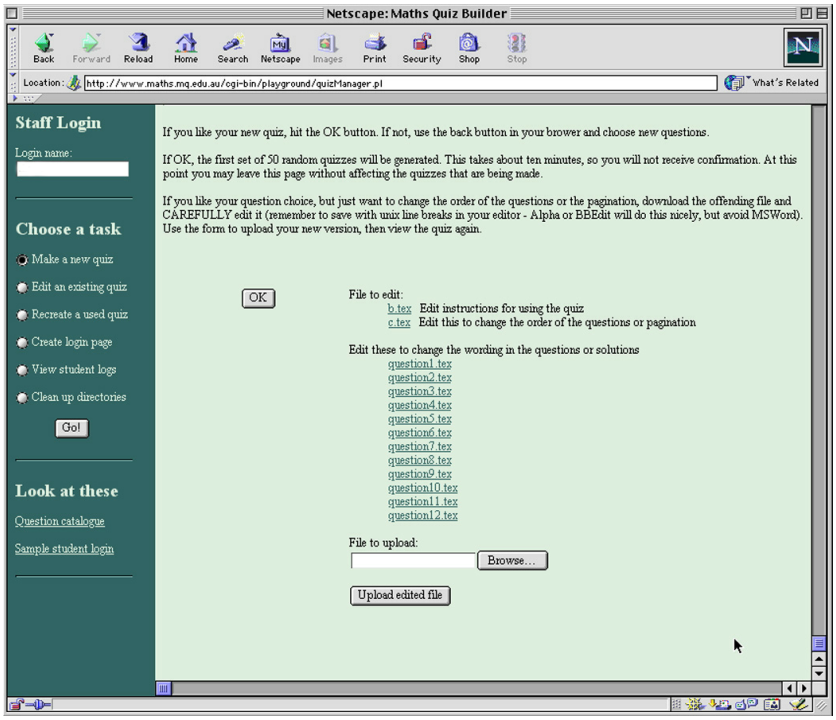

FIGURE 16. When a new quiz has been designed, this page provides access to the specific LaTeX source files [7] which control the appearance of each question. These files can be downloaded for detailed editing of the wording and page-layout, then up-loaded back to the appropriate directory.

SOFTWARE & PROGRAMMING LANGUAGES

For the MacQTeX quiz system to work, many different programming languages have been used to automate the various tasks. Here is a brief summary.

web interface: This is done using HTML, with some static pages and other pages (e.g. pages showing the class activity and individual student's log-files) being generated "on-the-fly" via CGI interfaces, currently implemented using *Perl* [10]. Similarly *Perl* is used to respond to any request to access a quiz document, for recording student information and access-details, as well as a selecting a unique quiz document to send via the web-server.

typesetting: The high-quality typesetting and layout of the PDF quiz documents is done using pdf-TeX [6], with the LaTeX [7] macros controlling most aspects of the page-layout. A modified version of the `exerquiz` [9] macros controls the specific layout of the quiz documents, and the inclusion of embedded JavaScript [1, 8] coding to control the interactive effects.

However, before any typesetting can be done, the complete job needs to have been carefully orchestrated. Dozens of files may be required to construct a single quiz. This is handled by *Perl* [10] scripts which, among other tasks, initiate the use of *Mathematica* [11] when randomisation is required, and collect together all of the resulting pieces.

randomisation: The uniqueness of different quiz instances is controlled using *Mathematica* [11], though other programs could be used instead. The requirement here is for a program that can do mathematical calculations as well as manipulation of character-strings, and be able to write the results directly into files as LaTeX source [7] capable of being understood by pdf-TeX [6].

web browser: Any web-browser can be used by students to access the quizzes, provided it has the appropriate Acrobat Reader [2] plug-in for reading PDF documents [4], or corresponding browser integration. (Alternatively, the quiz documents can be downloaded to disk and read as stand-alone documents, outside of a web-browser. However in this mode, there can be no submission of results back to the server for recording.) Similarly,

instructors can use any web-browser for the interface to the sites for creating new quizzes, and for examining the quiz-logs, or the personalised student log-files.

Further Development

The software as described above runs on a Sun Sparc Ultra 5, under the Solaris 5.7 Unix operating system. Thanks to support from the AUDF, we plan to implement everything also on a Macintosh OS X Server. Porting to MacOS X should not prove to be any great problem, as most of the software is already available for all Unix platforms.

Furthermore, we plan to build a new administrative application to run under MacOS X. This will provide an alternative interface for generating new quizzes and viewing log-files, which will not require web-based access. Since PDF is already the graphics engine for MacOS X, it is expected that some major parts of the quiz-generation system may be able to be reworked with other software, to allow greater flexibility in the nature of the quizzes that can be produced.

Biographical Information

Dr Ross Moore. Ross is a Senior Lecturer in the Mathematics Department at Macquarie University, where he has taught topics in mathematics, mainly with a significant computing aspect, for more than 10 years. His main area of research is in mathematical typesetting and developing ways to present mathematical ideas via the Internet. For this he has needed to master languages such as TEX, LATEX, *Mathematica*, PostScript, HTML, *Perl*, PDF, as well as some of the more traditional programming languages. Since 1997, Ross has been on the Board of Directors of the The TEX Users Group (TUG), an international organisation for the promotion and development of the TEX typesetting software and related programming methods.

Frances Griffin. Frances is a graduate student in the Mathematics Department at Macquarie University, completing a Masters degree in Random Number Generators and Cryptography. After training and working as a musician, she became interested in computers, and gained considerable experience in the graphic design industry, before returning to formal study. Her programming experience is quite extensive, covering languages such as Pascal, *Perl*, C, JavaScript, HTML, as well as using *Mathematica* and LATEX in her work. Frances also does part-time work as a tutor, and as an instructor in the Centre for Learning and Numeracy Skills at Macquarie University.

References

[1] Adobe Systems Inc.; "Acrobat Forms JavaScript Object Specification, Version 4.0"; Technical Note #5186; Revised: January 27, 1999.

[2] Adobe Systems Inc.; Acrobat Reader, viewer for PDF format [4] documents, available free of charge from http://www.adobe.com/.

[3] Adobe Systems Inc.; "FDF Toolkit Overview"; Technical Note #5194; Revised: August 10, 1999.

[4] Adobe Systems Inc.; "Portable Document Format, Reference Manual, Version 1.3"; March 11, 1999.

[5] Adobe Systems Inc.; "`pdfmark` Reference Manual"; Technical Note #5150; Adobe Developer Relations; Revised: March 4, 1999.

[6] Hàn, Thế Thành; pdf-TEX, free software for generating documents in PDF format, based on the TEX typesetting system. Available for all computing platforms; see http://www.tug.org/applications/pdftex/.

[7] Lamport, Leslie; LATEX, a Document Preparation System. This is free software available for all computing platforms. Consult the TEX User's Group (TUG) website, at http://www.tug.org/.

[8] Netscape Communications Corporation; Nestcape JavaScript Reference, 1997; available online at http://developer.netscape.com/docs/manuals/communicator/jsref/toc.htm.

[9] Story, Donald; `exerquiz` & AcroTEX, packages for including special effects in PDF documents, using TEX and LATEX. Dept. of Mathematics and Computer Science, University of Akron. Software available online from http://www.math.uakron.edu/~dpstory/webeq.html.

[10] Wall, Larry; *Perl*, a general purpose scripting language for all computing platforms. This is Free Software, available from http://www.perl.com/.

[11] Wolfram Research Inc; *Mathematica*, a system for doing Mathematics by computer. Consult the website at http://www.wri.com/.

Dr Ross Moore,
Mathematics Department, Macquarie University, Sydney
E-mail address: `ross@maths.mq.edu.au`
URL: `http://www.maths.mq.edu.au/~ross/`

Frances Griffin,

Mathematics Department, Macquarie University, Sydney
E-mail address: fgriffin@maths.mq.edu.au
URL: http://www.maths.mq.edu.au/~fgriffin/