emamo: An On-Line Self-Learning Tool for LP Modelling

Rasika Suriyaarachchi [*], Simon Dunstall [**] and Andrew Wirth [***]

Abstract

Operations research students find modelling challenging. This situation is not helped by the fact that traditional OR syllabi are often more focused on solution techniques rather than model building. With readily available software support, learning about solution techniques has, over the years, become less appealing to the student.

We report an attempt to help engineering students learn the craft of modelling by progressively guiding them from the problem statement to the final mathematical model by means of a web-based software system named ***emamo*** (for ***e***ngineering ***ma***thematical ***mo***delling).

* School of Computing and Information Technology, University of Western Sydney, Locked Bag 1797, Penrith South DC NSW 1797, Australia (rhsur@mame.mu.oz.au)

** CSIRO Mathematical and Information Sciences, Private Bag 10, Clayton South MDC 3169, Australia (Simon.Dunstall@csiro.au)

*** Department of Mechanical and Manufacturing Engineering, The University of Melbourne, 3010, Australia (wirth@mame.mu.oz.au)

Introduction

The aim of this paper is to introduce ***emamo***, a web-based tutoring package which allows students to develop and practise their skills in formulating linear programming models for decision making problems. The ***emamo*** name is derived from **e**ngineering **ma**thematical **mo**delling. The ***emamo*** application has been developed at the Department of Mechanical and Manufacturing Engineering at the University of Melbourne, Australia.

Anecdotal evidence suggests that operations research students find modelling challenging. Students perform poorly on modelling questions in exams. It is fair to expect a student to develop modelling skills through proper practice but some formal and intensive guidance is required at the beginning. Traditionally, this guidance was provided through tutors and mentors. However, with limited resources and the need for timing flexibility in the provision of such resources, it has become prudent to investigate avenues of providing flexible teaching / self-learning resources to supplement the formal teaching processes.

The web-based tutoring package, ***emamo***, was developed with the objective of providing such a supplementary learning resource for Linear Programming (LP) modelling.

In this paper, we first provide some background information about the teaching environment that led to the conception, design, development and implementation of ***emamo***. The design and development process is then described along with a description of the salient features of the package. We conclude the paper with other important details about the implementation of ***emamo***, the student feedback received, and about future improvements to the system.

Background

Linear programming is taught at the Faculty of Engineering in the University of Melbourne as part of an engineering mathematics program. Many undergraduate engineering students take the core subject Engineering Analysis B in their second year where an introduction to operations research constitutes about one sixth of the subject load.

In the minor module on LP, students are introduced to the concepts of building mathematical models to represent various decision problems found in the engineering profession with an emphasis on LP modelling. Students are then introduced to the simplex algorithm. Numerical solutions are usually generated using *MATLAB* software.

It has been observed over the years that students find it comparatively difficult to grasp the concepts associated with formulating LP models. This is reflected in the answering patterns of the final examination where with a choice given, students prefer answering alternatives to questions related to LP modelling.

Most of the Engineering Analysis B subject requires students to apply standard techniques such as solving the characteristic equation in order to find the solution to a given higher-order differential equation. The section on LP modelling is unusual in that it requires the student to *construct* the model rather than to solve a problem which as already been modelled. As is well-known, students typically prefer applying a 'cookbook' approach which involves use of some standard technique, to having to answer a seemingly much more open-ended question. After one lecture, a very agitated student complained that "...*this modelling is like creative thinking*…".

Traditionally, students are expected to practice by trying modelling problems given in lecture notes and texts with help and guidance available through tutorials. However the time allocated for both lectures and tutorials is limited. Furthermore, with large student numbers and a high variability in student learning abilities, it is not practical to provide tailored tutoring support during the tutorial classes. Hence, there is a pressing need for a supplementary teaching / self-learning resource for LP modelling.

It was proposed to develop a web-based interactive tool for this purpose and ***emamo*** is the outcome of that proposal.

Development of *emamo*

The ***emamo*** application is an on-line self-leaning tool designed for access via the World Wide Web. There are a number of reasons why ***emamo*** was developed as a web-based tool.

The current trends in the provision and the delivery of educational services indicate that the web is going to play an ever-increasing role in the future. Major education initiatives such as Universitas21, and proposals by major political parties, are centred on the Internet.

There are quite a few benefits associated with delivering a tool like ***emamo*** over the web. Firstly, the web has become one of the most familiar user interfaces and almost without exception each and every member of the target audience is quite familiar with it. The web is also (nominally) platform independent, that is, , ideally the applications can be made available on Windows-based PCs, Apple Macs and various unix/linux workstations alike. However, true platform independence for web-based applications is yet to be achieved in practice.

The delivery of the tool is rather easier over the web, at least in concept, as maintenance and upgrades need only be carried out at the server. In the case of ***emamo***, new problems can be added without individual distribution among the users. Perhaps the most valuable benefit of using the web is the ease in accessibility. Students are able to use the application whenever and wherever they wish to do so.

Funding for the development of ***emamo*** was provided by **TaLMET**, the Teaching and Learning Multimedia and Educational Technologies project grant scheme of the University of Melbourne. Responsive communication between user computers and ***emamo*** is enabled through use of OCCA (On-line Coursework Component Architecture) developed by the Multimedia Education Unit (MEU) of the University. The authors acknowledge the support of MEU, for their assistance and in hosting the development version of ***emamo*** and its database.

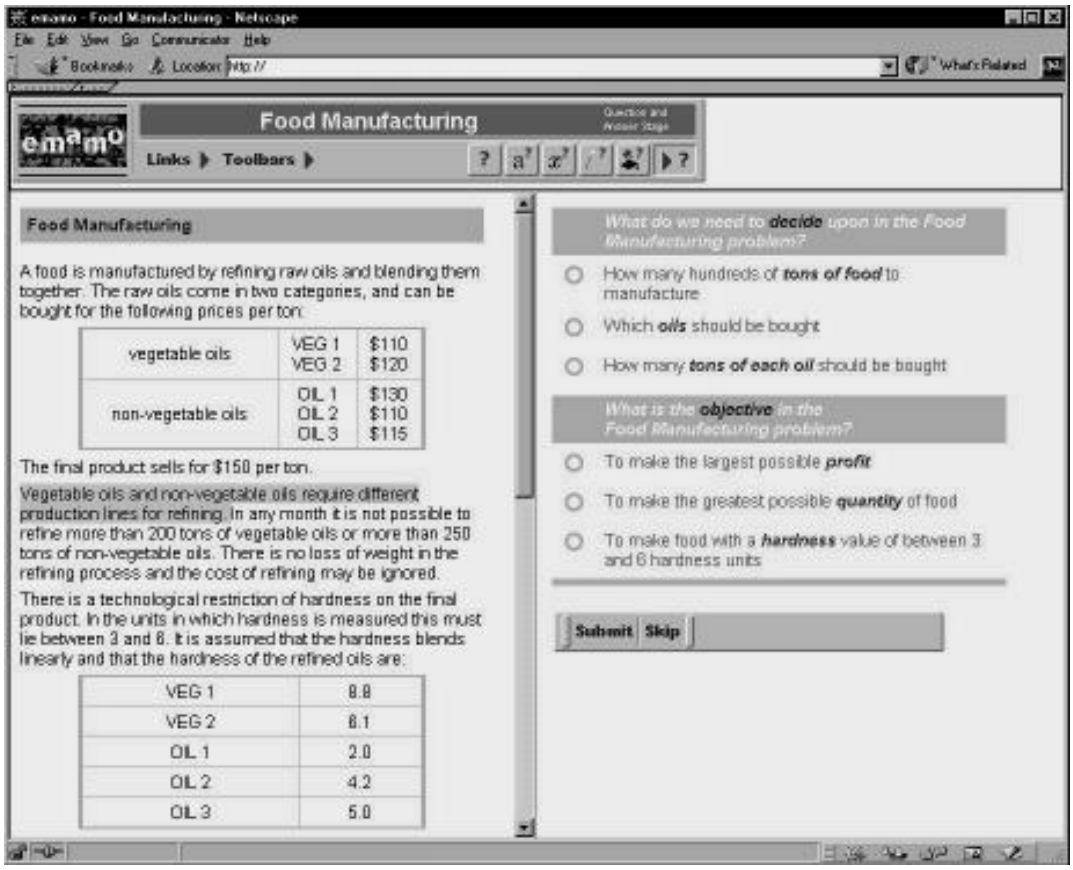
Features of *emamo*

emamo is designed to provide a tutorial-like experience. It contains a series of problems inspired by real-life case studies from various areas of engineering and management. Each of these problems can be formulated as a linear or mixed-integer program.

Users first have to log into ***emamo*** before proceeding to the problem they choose to work on. They then go through the modelling process, from interpreting the problem to completing the LP model. Users can work at their own pace, go backwards and forwards as they wish, and leave ***emamo*** at any point to continue work at a later stage.

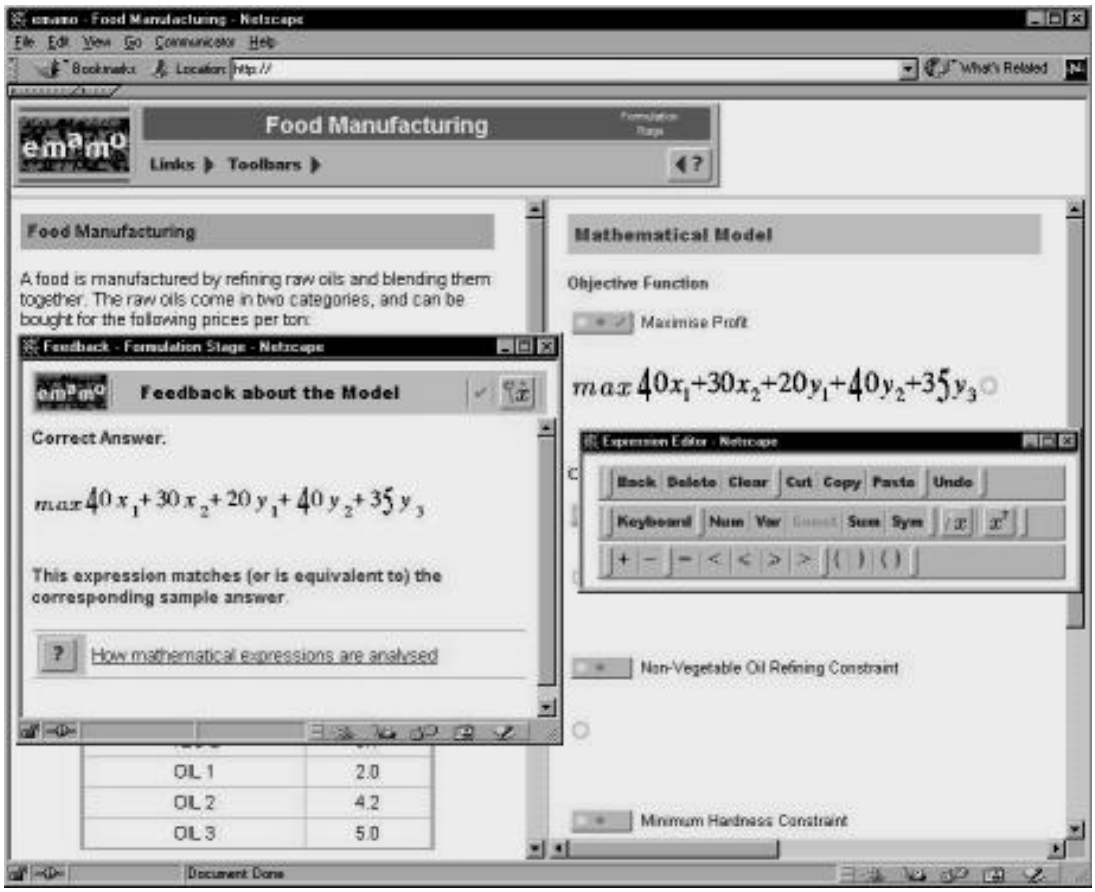
The user interface of ***emamo*** consists of three separate sections as seen in Figure 1. The section that occupies the top portion of the screen contains the title of the problem being worked on along with a collection of toolbars. The remaining part of the screen is divided into two sections. The left section contains the problem description with text and graphics, and remains largely unchanged during the whole exercise. The right section of the screen is used to prompt and interact with the user. In addition to the main browser window, ***emamo*** uses several other pop-up windows in order to communicate with the user.

Figure 1: The user interface of *emamo*



A typical ***emamo*** session has two stages. In the (first) *question and answer* stage, the student is asked to identify (through a series of questions) the *objective function*, *decision variables* and the *set of constraints* required to model the problem as an LP. Commonly this process involves the selection of an answer from of a set of options. When the student makes his/her choice and submits the answer, ***emamo*** provides appropriate *feedback* using a pop-up window. To help the student to make his/her selection, *hints* with descriptive background material are provided for each paragraph of the problem and for various components of the questions. These *hints*, too, are delivered in a pop-up window.

Figure 2: Building a linear programming formulation with *emamo*



Once the objective function, decision variables and constraints are identified, the user is asked to link these specific parts of the LP models to the relevant parts of the problem text. The aim of this interactive segment is to further the student's understanding of the problem. At the end of the question and answer stage, the student will have selected an appropriate notation and a set of decision variables for use in formulating the problem.

In the (second) *formulation* stage, the student progressively and interactively enters the LP formulation. This process is facilitated through use of a graphical *expression editor*, complex expression-checking techniques for matching student answers to

standard answers, and a series of graphical indicators and textual information elements that respond to various actions initiated by the student (see Figure 2). All of these components have been developed specifically for this project. Both a keyboard-driven and a point-and-click approach to entering mathematical expressions (that is, the objective function and constraints) are available to the student.

Mathematical expressions can be checked for correctness either individually or as part of a completed formulation. A student initiates the expression-checking process by clicking on function buttons or graphical indicators. In case of an erroneous expression being entered, the feedback provided by ***emamo*** will guide the user towards the correct expression by indicating (for example) that there are variables missing, unwanted variables, non-linear terms or incorrect coefficients of variables.

At any point of interaction with ***emamo*** the user can store all information relating to their current position within the application, as well as data about their progress, in a database at the ***emamo*** server. This storage process is invoked either directly by the user (by clicking on a *save* or *submit* button) or indirectly during the process of departing a page (via a link or some other common navigation feature). This way, users can return to ***emamo*** at any time and continue from where they were up to.

The vital client-server interaction to achieve data persistence is mediated by a Java applet at the client and various scripts at the server. All other dynamic user interaction and data processing is currently undertaken client-side (that is, at the student's computer) using a large array of JavaScript objects and functions.

In addition to a generous supply of detailed *hints*, *feedback* and generic *help* information embedded in ***emamo*** (as a mixture of static and dynamic HTML content), during the formulation stage ***emamo*** also makes available to students the option of asking ***emamo*** to directly provide the mathematical expressions. For this purpose a feature entitled *magic* is included in the application.

The *magic* feature can be used to insert complete expression(s) into the formulation if and when the user decides that he/she is simply not able to compose certain parts of the formulation independently. The *magic* function may be disabled by the lecturer, as required.

The ***emamo*** application monitors the use of features such as *hints* and *magic*, and will supply a tally of their use to a student. The student has the ability to reset this tally, and to revise and restart any problem module. The above is in keeping with a key ***emamo*** objective of handing students complete control and responsibility for both their progress through the system and (most importantly) improvements in their modelling skills.

Feedback on *emamo* Usage

A small number of students were invited at a very early stage of development to comment on the user interface. They suggested that the problem statement be continually displayed on the screen and we have incorporated this feature in all of the problems. A further small group of first year students were asked to work through one of the problems (the "supply chain problem") and were given a bare minimum of

hands-on support in doing so. They were successful in constructing the appropriate model and along the way identified a series of bugs and quirks in the application.

Since then ***emamo*** has been incorporated into the curriculum of the Engineering Analysis B subject as a supplementary teaching tool. The students for this subject will be requested to provide formal feedback by means of a questionnaire towards the end of the semester. This and other forms of student response to ***emamo*** will be used to identify areas for the future improvement of ***emamo***.

Further Developments of *emamo*

The concept and appearance of the ***emamo*** user interface has remained largely constant since the product started to take shape in early 1999. However, at the internal level, ***emamo*** has now undergone several major phases of development since the completion of an initial proof-of-concept version that contained a single problem module and a subset of the UI functions.

emamo now contains five modules (problems) selected from various areas of engineering and management. The addition of further problems is a continuing priority. One of the significant reasons for this priority lies in the variation of engineering domain knowledge across the target group of students. From teaching experience it is known that familiarity with the problem context (for example, familiarity of a mixing problem to chemical engineers, or a production-planning problem to manufacturing engineers) can greatly increase the rate at which students grasp the underlying mathematical and model-formulation concepts.

Improvements in web browser capabilities (for example, MathML) present opportunities for revision and enhancement of the ***emamo*** application. We are particularly aware of the limitations of our current methods for entering and displaying mathematics, and we will re-visit these in the near future. However, the most important input to further development processes will be the student feedback that we will gather towards the end of the current teaching semester.

Conclusion

In some sense this project is still very much at a formative stage and so any conclusions we may draw would be somewhat premature. Nevertheless it is clear that with the limited resources typically available to such projects their development will inevitably be very protracted. Furthermore we have had to lower our sights as far as the number of case studies to be included. We were initially aiming for twenty-five or so: we now have five covering manufacturing, telecommunications, investment and project management.

The issue of the efficacy of the program is a vexed one. We have sought expert assistance with the evaluation of the project and as a result have decided to distribute questionnaires asking for qualitative student comments. Ideally it would have been useful to compare the performance of students who used ***emamo*** with those who did not. However, there were too many confounding factors to make this feasible. The project will also be peer reviewed.

A preliminary study of the literature has not revealed any similar developments elsewhere. Perhaps the closest one is tutOR, developed by the Department of Mathematics and Statistics, University of Melbourne. tutOR covers a much wider range of topics but emphasises algorithmic solutions rather than model development.

Modelling skills are developed over a long period. Past Engineering Analysis students have found this part of the course relevant and interesting but very challenging. Hopefully emamo will aid their learning process. It should also be quite possible to adapt ***emamo*** to other modelling courses. We hope to report on these developments at a later date.