Fourth International Derive TI-89/92 Conference
Liverpool John Moores University, July 12 – 15, 2000

MathsWeb: An Intelligent Computer Algebra System for the World Wide Web

Dhiya AL-Jumeily and Paul Strickland
Liverpool John Moores University, UK
Email: D.Aljumeily@livjm.ac.uk, cmspstri@livjm.ac.uk

Abstract
This paper describes an Intelligent Tutoring System for algebra. The system is called "MathsWeb". One of the primary aims of the system is to be used as training environment in school classroom as well as home, therefore both school children and teachers of mathematics have been involved throughout the life cycle of MathsWeb. The design of MathsWeb has been based on the results of school experiments that was conducted at schools and the resulting product was evaluated by school children.

MathsWeb incorporates techniques from Intelligent Tutoring Systems. In particular, it takes a reconstructive approach to student modelling and error diagnosis, which uses a generic model of the domain for locating as well as identifying the reason of the error, in each step of student's solution.

MathsWeb has been evaluated by school children. The results of the evaluation show that the system has a positive effect on the learning process. In other words, the system has performed better than the traditional learning environment.

1. Overview of MathsWeb

MathsWeb is an intelligent computer environment developed to support algebra manipulation skills. The system can be considered as a black/white box model in the sense that it can be used to check any intermediate step within the student's solutions, as well as the final answer. The system was designed to be close to using pencil and paper, so that it can be used to support the traditional method of the teacher presenting the method to the student and asking them to practise what they learned by doing some exercises. As the student uses the system, it has the capability of providing immediate feedback within any step of the student's solution. This is believed to help the learning process, as discussed in the evaluation study. An optional explanation is available with the system, for a student who wants to know more about their error. However, there is no requirement that steps should form part of a teacher's "model solution", allowing the pupil to experiment freely while developing their skills. In this sense, MathsWeb can best be described as an assistant tutor. Though it will never completely replace the human tutor, MathsWeb can and does offer to assist the human tutor by addressing and reinforcing the generic techniques of algebra manipulation, which need to be mastered by the students as part of the curriculum. This will leave the human tutor free to spend more time with those in need of individual help.

2. Overall Design of MathsWeb

There are different design criteria that must be met to provide a successful computer environment for education in algebra. Following Beeson's work (Beeson 1989; Beeson 1992; Beeson 1998) and for the purpose of developing the MathsWeb system, this section is intended to illustrate some design principles that can be used as guidelines for designing any such computer system. These design principles involve mainly the internal operations of the software. The design of the interface

is also critical, and much has been written about Graphical User Interfaces (GUIs) and the human factors, which should be taken into consideration. The design methodology for MathsWeb followed the guiding principles outlined in the literature (Macaulay 1995; Dix, Finlay et al. 1998).

2.1 Purpose of the Software for Mathematics education

The first step in good software design is to clearly state the purpose of the software. So, MathsWeb (Al-Jumeily and Strickland 1998) can be defined as an intelligent computer algebra system, developed for the purpose of replacing the traditional paper-and-pencil method of learning algebra, with an intelligent training ground for practising tactical algebraic skills. In other words, the system provides the student with a "learning-by-doing" environment. Most educational experiments to date using algebra software utilise systems not developed specifically for education but for use as black box tools (Beeson 1998). This means that their capabilities may not meet the educational needs of the students. The lack of learning support motivated the development of the MathsWeb system toward the educational needs of the students.

2.2 White Box / Black Box Principles

Domain expertise in a computer system can take one of two forms. One form is called “glass box” or “white box”, which means that you can see how the computer system arrives at its answer (Anderson, Boyle et al. 1985; Beeson 1998). The system in this case allows the student to construct, or present step-by-step solutions, not just the final answer. The model is referred to as “articulate” or “cognitively faithful”, if its own internal solutions corresponds to the solutions a human could produce (Anderson 1992). The second form, called the black box, has data structures and processing algorithms, which are hidden, and do not necessary mimic those used by human beings. Systems of this type have the capability of giving an answer to a problem without giving any intermediate steps of the answer, i.e. one-step solution.

According to Buchberger (Buchberger 1990), MathsWeb can be considered as “white box/ black box”, since it has the capability of checking step-by-step answers that model student solutions. As the student types a solution, the system monitors the student’s performance step-by-step, providing feedback on errors to help student to put their answer right. However, it can also be considered a black box system since it has the capability of accepting a one-line solution. MathsWeb supports this paradigm fully allowing the user complete flexibility to customise the level of detail desired in the solution. Figure (1) shows a comparison of two student solutions, so the beginner algebra student can solve some thing like *4+(x+2)(x-2)* in about 4 steps, whereas an advanced algebra student can solve the same example in 2 steps.

Beginner		***Advanced***	
$4+(x+2)(x-2)$		$4+(x+2)(x-2)$	
$4+x^2+2x-2x-4$	*Multiply Brackets*	$4+x^2+2x-2x-4$	*Multiply Brackets*
$4+x^2-4$	*Cancelling*	x^2	*Cancelling*
x^2-4+4	*Rearranging*		
x^2	*Cancelling*		

Figure 1: Sample step-by-step solutions

2.3 Immediate Feedback

MathsWeb is different in many respects from well-known software systems (for example, MACSYMA, REDUCE, MAPLE, MATHEMATICA), which we believe will have implications for

education. One of the most important aspects is that it checks the correctness of student answers step-by-step, and can detect the majority (we hope) of mistakes made in these steps. Most importantly perhaps, MathsWeb also has the power to provide immediate feedback about errors. It should be noted however, that feedback does not display problem solutions, it merely specifies errors (with an explanation of a problem that may have caused these errors). Two types of feedback are provided by the system, the first type simply tells the user an error has been made, while the second type of feedback is an explanation of an error. The explanation feedback specifies the error within a step of the student's solution. The error within the incorrect term is flagged in red accompanied with an explanation of reason/s that could have caused the error/s. Full details are presented in chapter six of this thesis.

2.4 Ease of Use

One of the points that have been emphasised for many years in the development of mathematical educational systems is that the system has to be easy to operate. Students are often afraid of two main things, computers and mathematics. So it is essential that the students feel that the computer is a help, not just another thing they have to learn, which in this case will add to their difficulties, rather than helping them. Agreeing with Beesen's argument (Beeson 1998), ease of use is not an interface issue only, since well designed educational software should not be separated completely into an interface and a kernel. The ease of use relating to the interface is discussed in chapter six of this thesis; this section is primarily concerned with the kernel. It could be claimed that the system is easy to use since it checks the answer step-by-step rather than the final answer only, which in this case is a kernel issue. Another related point is the location of the student's error, which will be pointed out to the student. This issue could be considered as mixed issue of both interface and kernel.

2.5 Support for Classroom Learning

MathsWeb is a software system in mathematics designed explicitly to support the learning of algebra. In the recent years there has been much interest in incorporating computers into the teaching of secondary and introductory university level mathematics, particularly in the use of computer algebra systems in introductory calculus and algebra. Existing computer algebra systems, such as Mathematica and Maple, have been designed primarily to assist experts in doing mathematics, rather than as an aid for teaching mathematics. This has been found to have several adverse consequences (Hayden and Lamagna 1998). First, students must learn an unfamiliar language in order to interact with the machine. Second, these systems are designed to provide answers and not to show how these solutions are obtained. Another difficulty is that existing algebra systems tend not to provide the kind of operations that students employ in stepping through the solution to algebraic problems. In other words, these systems do not support learning for the requirements of standard curriculum mathematics, which emphasises step-by-step solutions. This has led some educators to call for a new curriculum, in which students would not learn the traditional step-by-step solution methods (Delozanne, Bruillard et al. 1992). If we do not want to change the curriculum, then we need to provide specific software that can be used as a support for the existing curriculum (Beeson 1998). MathsWeb has been designed with this in mind to provide step-by-step review of the student's solution. At the same time, the human tutor has the capability to edit the question, so it could be adapted to different levels of the school curriculum.

3. Functional Facilities and Architecture of the MathsWeb

MathsWeb is an intelligent CAS built purposely to support education. The domain of expressions that can be manipulated is currently limited to integer polynomials. The architecture of MathsWeb is determined by its functional facilities and some additional design requirements like transparency and explanation generation capability. Okamoto in his paper (Okamoto 1992) has pointed out that one of the important issues in the process of developing any system is specifying the functionality which satisfies the aims and objectives for building that system. In the light of this, it is suggested that the following functions must be provided with the system:

- A function to identify the type of student error (logical error or parse error).
- A function to check the equivalence of two expressions (one step in the student's solution).
- A function to locate the error within the student's solution.
- A function for constructing an explanation for the error.

The architecture of MathsWeb is shown in figure (2). The detailed functionality of each part of the system can be found in the following sub sections.

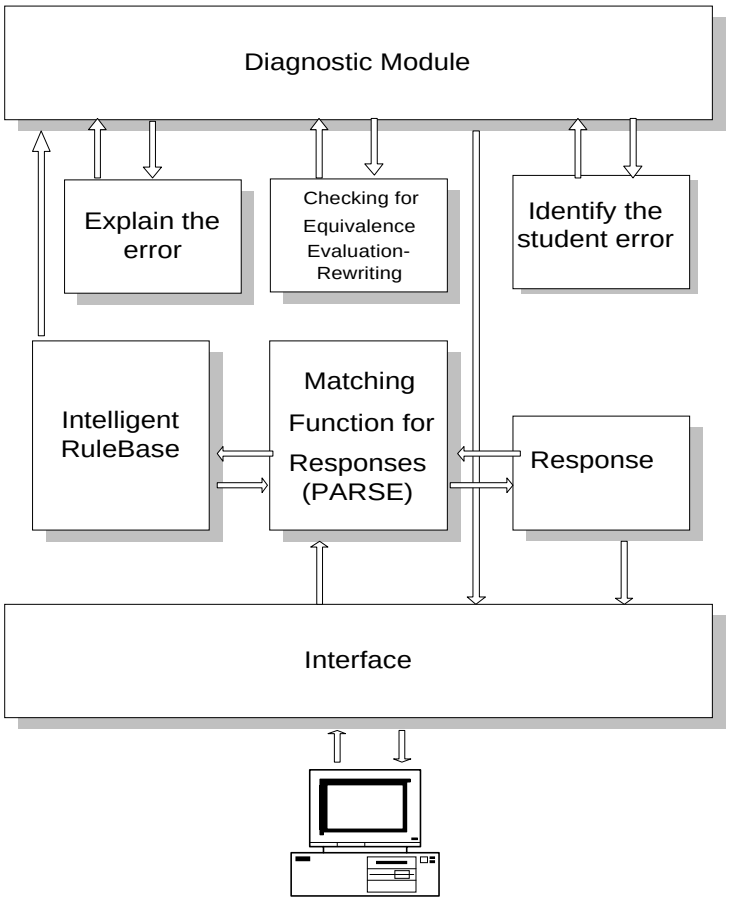

Figure 2: The Intelligent Computer Algebra System configuration

3.1 Interface/ Response

When the expression is input by the student, a function is required which recognises the student's response and presents an adequate message according to the student's answer. There are three

functions available with the system, to provide an adequate message to the students, which are the following:

The first function is to recognise a typing error, which gives a message to urge the student to re-input the expression. The second function is activated upon recognising an error (which is not a typing error). Both functions have the capability to allow the student to edit their input.

The third function is used to identify when the student's input is correct but it is not the final answer. This function will encourage the student to input another step, leading to the final expression. However, if the system detects that the submitted expression is the final expression, then the system congratulates the student, with the message 'Correct, well done'.

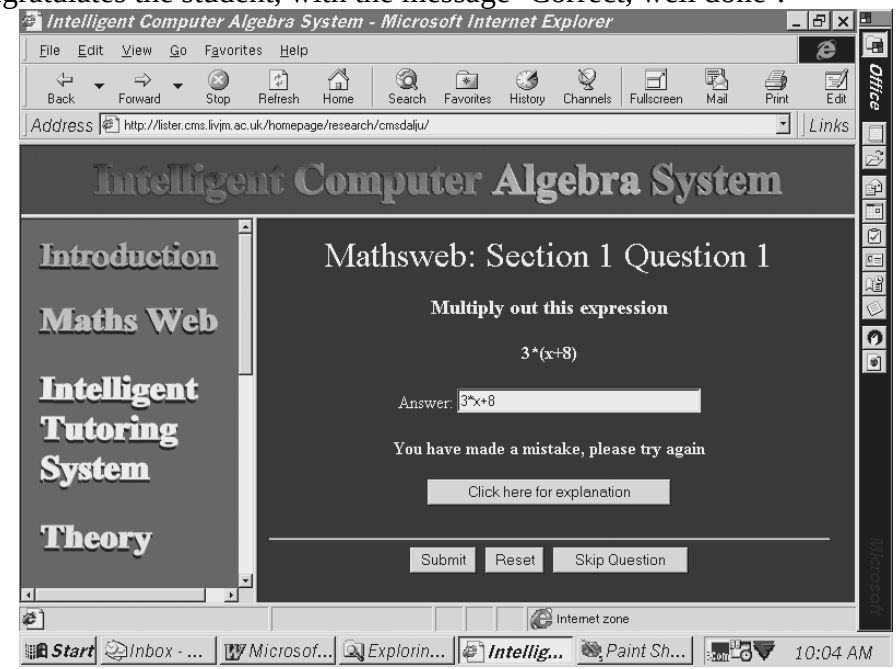

Figure 3: A screen shot of the Intelligent System

The interface of the system has been developed in accordance with the principles of good graphical user interface (GUI) design. Full details of it can be found in (Al-Jumeily 2000). The first stage of the system is the logon window interface enabling the student's data to be recorded. This will lead the student to another screen that asks the student to answer a set of questions. A typical screen shot of the system can be found in figure (3).

3.2 Intelligent RuleBase

The system contains solution knowledge to modify the mathematical expression, by the method of rewriting. The knowledge is the technique provided by the rules to transfer a term (mathematical expressions) to another equivalent term, using rewrite rules. The structure of each rewrite rule includes the description of a correct transformation of a class of mathematical expressions and the formulation of some general preconditions for its performance (if there are any). A full description of term rewrite rules can be found in chapter two of this thesis. The rules include not only the ordinary rewrite rules, but also include other rewrite rules, which are organised into several sets, namely, transparent rules (rules that are expected to be known by the student), mal-rules and explanation rules.

The first set of rewrite rules is the ordinary rewrite rules, which includes two types of rewrite rules, regular rewrite rules, and conditional rewrite rules. Table (1) presents a set of rules that are sufficient to simplify a linear expression with one unknown, specifically for the algebra domain of integer polynomials.

The conditional rewrite rules arise from the fact that some mathematical laws are not universally valid, such as $nx = m \rightarrow x = m/n$, which is only valid when n is non-zero. Another example is the binomial theorem, where only non-negative integral exponents give us universally valid equalities; and other exponents lead to infinite series. For such special circumstances, a set of rewrite rules have been constructed, such as $X^N \rightarrow X * X^{(N-1)} if\ N > 0$. If this rule could apply to y^0, the following $(y * y^{(-1)})$ could be obtained, leaving the domain of integer polynomials; and, even worse, this expression could be rewritten infinitely often in terms of y^{-2}, y^{-3} and so on.

The second set of rewrite rules is the set of transparent rules, which can be defined as a basic algebra rules that should be well-known by the students. This type of rules are dependent on the subject matter being taught, for example $-(-x) \rightarrow x$, and $x + 0 \rightarrow x$. This is considered to be a set of rewrite rules that need to be well understood by the students before they are exposed to more algebra manipulation rules, and the system will execute these rules immediately.

The third set of rewrite rules are the incorrect rewrite rules (mal-rules), which can be used to express the errors that students make. One must decide on a complete and consistent set of concepts and operations in the domain, and determine possible difficulties that students have. In the multiplication of term by brackets, a sample test could include multiplying a constant by bracket (of two added terms). Some students multiply with the first term but not all of the terms of bracketed expressions. It has been observed that some students produce the answer $2 * x - 2$ to the question (simplify the following expression: $2 * (x - 2)$). This student answer could be modelled by the following mal-rule: $X * (Y + Z) \rightarrow X * Y + Z$ where X, Y and Z stand for any expression in the given domain. A full description of these rules can be found in (Al-Jumeily 2000).

The final set of rewrite rules is the explanation rules. The purpose of these rules is to provide explanatory feedback to the students, about their errors. Full details of these can be found in a later section of this paper.

Correct Rules	
Rules	**Semantics**
*A^N ⟶ A*A^P* ***where*** *(P* ***is*** *N-1 and N>0)*	*Rule of exponentiation*
*N*A+M*A ⟶ P*A* ***where*** *(P* ***is*** *N+M)*	*Addition of unknowns*
*N*A+A ⟶ P*A* ***where*** *(P* ***is*** *N+1)*	*Addition of unknowns*
*A+A ⟶ 2*A*	*Addition of unknowns*
A(B+C) ⟶ A*B+A*C*	*The distributive law*
(A+B)(A+B) ⟶ A*A+2*A*B+B*B*	*Expanding brackets*
Transparent Rules	
Rules	**Semantics**
0+A ⟶ A	*Adding zero to unknown*
+(A) ⟶ A	*Positive sign*
*1*A ⟶ A*	*Multiplying one by unknown*
*0*A ⟶ 0*	*Multiplying zero by unknown*
-A ⟶ (-1)(A)*	*Taken the negative sign as (-1)*
A - B ⟶ A+ (-1)(B)*	*Taken the negative sign as (-1)*
A^0 ⟶ 1	*power zero of any unknown is one*
Mal-rules	
Rules	**Semantics**
*A*B ⟶ B*	*Delete a multiplicative factor*
A+B ⟶ B	*Deleting an additive term*
Where *M*, *N* and *P* are Integer. *A* , *B* and *C* are polynomial expressions. (***is***) used as standard PROLOG built-in arithmetic, which can be used to evaluate ground terms.	

Table 1: Example of the rewrite rules written in PROLOG format

3.3 Checking for Equivalence

This function is used to check a single step in the student's solution. This consists of two expressions (student's present input and student's previous input). In the case of the first step, the previous input is represented by the original expression. This function uses the complementary techniques of rewriting and evaluation (described in (Al-Jumeily 2000)), for checking one step in the student's solution. The intelligent algebraic system makes a single substitution randomly for the variables to check whether the two expressions are apparently equivalent; if they are, then the rewrite rules (both ordinary and transparent) are used to confirm equivalence. If the two expressions are found not to be equivalent, then the system attempts to find an explanation for the error, as well as locating the error with the student's solution.

3.4 Identifying an Error

If the two expressions in the student's step are found not to be equivalent, then the system attempts to detect the error/s within the student's solution. As with the previous sections, each step of the student's solution will be represented as terms, say T_1 and T_2. These terms will be subject to term rewrite system, and in this case for two sets of rewrite rules (the correct rewrite rules (say R) with a rewrite relation $\rightarrow$, and the Mal-rules (say M) with a rewrite relation $\overset{\circ}{\rightarrow}$). In addition, the third set of rewrite rules (the transparent rules) will be implemented automatically. The system defined by R must be confluent, and that the combined system $M \cup R$ must be locally finite and terminating (since a system including erroneous rules need not be confluent). The set of mal-rules respects the polynomial ordering because the polynomial ordering has subterm property (each mal-rule replaces term by a subterm). The two terms T_1 and T_2 are rewritten into a common form, using the valley algorithm (Al-Jumeily 2000). This will initially consist of a highest valley between the two terms, by which it means a term V such that; [1] $T_1 \overset{*}{\rightarrow} V$, $T_2 \overset{*}{\rightarrow} V$ and such that there is no other term U such that $T_1 \overset{*}{\rightarrow} U$, $T_2 \overset{*}{\rightarrow} U$ and $U \overset{*}{\rightarrow} V$. This model should be found quickly, so feedback can be given to the student. To understand the procedure of specifying the error better, consider the following example;

> Suppose R is a standard rule system for polynomials over the integers, having rules for the distributive law and gathering of terms, and M consists of the following two mal-rules;
>
> $X + Y \overset{\circ}{\rightarrow} X$ {Deleting an additive term}
>
> $XY \overset{\circ}{\rightarrow} Y$ {Delete a multiplicative factor}
>
> These mal-rules can be used to join any two expressions having a common subterm (such as variables), so some sort of error model will be possible in almost all cases.
> Now assume a student enters the following simplification step;
> Previous step: $x - 5(x + 5)$
> Present step: $x - 5x + 25$
> The previous step can be reduced by R to ($x - 5x - 25$). Then the first mal-rule can be applied to two terms, to delete –25 and +25 from the two terms, arriving at an identical term ($x - 5x$) as shown in figure (5.4). This will indicate that there is a problem with (+25 and –25) in the two terms, which will be accompanied by indicating that a sign error is a likely explanation of the mistake.

[1] $S \overset{*}{\rightarrow} T$ *the notation denotes the fact that a term* S *can be rewritten to a term T by* $\rightarrow \cup \overset{\circ}{\rightarrow}$ *in zero or more steps.*

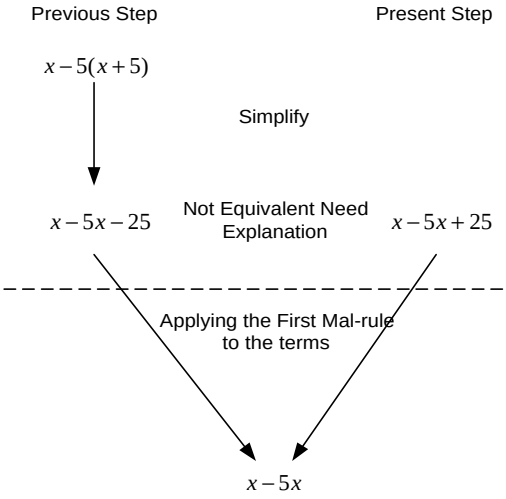

Figure 4: Applying mal-rules as a way of locating the student error

This technique has been used for the development of some components of the intelligent computer algebra system, which are the following;

- Modelling the feedback of the student's error/s.
- Developing the explanation part of the system.

It will also be used in the future development of the student model part of the system.

4. System Development

MathsWeb is developed for the purpose of examining issues concerning the effect of on the learning process. One of the system's distinctive characteristics is that it provides immediate feedback after the student types each step of his answer.

If the student makes a mistake, the system immediately interrupts, provides a message to the student "you have made a mistake, please try again" and requires that the student correct the error before moving on. If the student demands a clarification of the error, then the system can provide an explanation of the error. This explanation includes both identifying the error and locating it.

The effect of the feedback procedure is to prevent floundering by keeping students on a correct solution path.

As a result of the teaching experiment, a new version of the MathsWeb has been developed. In the new version, it has been planned that the system can notify the student immediately when he makes mistakes, by providing feedback (identifying the error and locating the error). However, the system does not provide feedback messages unless requested but does ask the student to repair the error. In MathsWeb, the following procedures are to be followed:

- Students are constrained to type their answer to each step, are informed immediately about the correctness of their answer, and are required to repair errors before moving on.
- The system provides an explanation if the student asks for it.

4.1 How the System Presents Feedback

At present, the system does not create a model of the student. The system uses a model of expert performance by showing how the students actions fit into a framework that the expert uses to evaluate them. It is assumed that students, supported by the teacher (through immediate feedback), can carry out the interpretative work required to form the expert concept based on the information

provided by the system. In other words, MathsWeb presents an interpretation of the error, rather than explanation.

In this section, a technique has been developed for the purpose of producing an interpretation of the error. One of the major problems facing the development of computer aided assessment and intelligent tutoring system is providing feedback to both the teacher and the student, which will involve identifying the error, locating the error and providing an interpretation of the error (which algebra rule has been mis-used).

4.2 Detecting the Error

A system to detect the error in the student's solution has been realised through the use of rewriting techniques over a set of both rewrite rules and mal-rules. The idea here is to check one step in the student's solution. This can be done by comparing the previous and current expressions in a step of the student's solution. If the two expressions are found to be equivalent (through evaluation and rewriting) then we can confirm that the student's answer is right. However, if they are found not to be equivalent, then this means the student's answer is not right, and more explanation is needed, as shown in the next section.

4.3 Identifying and Locating the Error

A combination of rewrite rules and mal-rules can be used to identify as well as locate the error. Mal-rules in this case can be used to join any two expressions having a common subterm (such as variable), so some sort of error model will be possible in almost all cases.

Consider for instance, the simple distributive law $X*(Y+Z)=X*Y+X*Z$. A typical student error will be multiplying X with the first term of the bracket but not with the rest of the bracket, for example, the student could write $X*Y+Z$ as an answer for the above example.

The first step to provide feedback to the student is identifying the error. A combination of both rewrite rules and mal-rules will be used to identify the student's error. This usually produces several possible answers, since there are different ways of simplifying any expression using a collection of rewrite rules. At first of developing MathsWeb, a typical answer from the above example was:

Term is missed from the original expression: x*(y+[z])
Term is added to your answer: x*y+[z]

Highlighted in here with square brackets around the terms. This means that *z* been removed from the original expression and added to the student's answer, which represents an identification of the student's error. However, presenting this to the student as a feedback has been found to be confusing. This was the finding of one of the teaching experiment carried out as a result of this research work. The experimental results have shown that the children with no feedback from the system performed better than the children with feedback from the system. The researcher has observed a confusion among the children because of the messages that the system produced. This could be the reason for the poor performance of this group.

Suppose the student been asked to simplify the following expression $x*(x+3)-7*(x-7)$, and as a result, the student enters the following expression $x^2+3x-7x-49$. The original expression can be reduced by the set of rewrite rule *R* (presented in chapter five) to $x^2+3x-7x+49$. Applying the

one mal-rule ($X + Y \rightarrow X$) to each of the two expressions will delete the +49 and -49 from the two sides. A typical answer from the system will be:

Term missed from the original expression: $[-49] + (-7 * x) + 3 * x + x * x$

Term added to the student answer: $[+49] + (-7 * x) + 3 * x + x * x$

As we can see, it is difficult for the student to make sense of it, if it is provided as an explanation.

An attempt has been carried out to model the numerous feedback provided by the system for the purpose of making suitable feedback. This will be in a form that identifies the error and its location in a way that can be easily understood by the student. A technique called *remove-out* has been developed for this purpose, which is presented in details in the next section.

4.4 The Basic Remove-out Tactic

A *Removal* is a term from which a proper subterm is deleted. In a similar fashion to *rippling* (Bundy, Harmelen et al. 1990), the idea of *remove-out* is to move the removed function so that it becomes the outermost part of the expression. The idea here is modelling a set of rewrite rules (explanation rewrite rules) that can be used to improve the form of the identification of error that presented to the student and model a better explanation of the error for the student. Such a rule will push the remove function to the outside of the expression. In the proposed system, the following explanation rules have been implemented:

$X * ([A] * B) \rightarrow [A] * B$
$X + ([A] * B) \rightarrow [A] * B$
$X + ([A] + B) \rightarrow [A] + B$

Suppose the student has been asked to simplify the following expression; $2 * (x + y)$, and as a result the student has entered the following expression; $2 * x + y$. It is clear from the student's answer, that the student forgot to multiply "y" by "2". In this case a combination of rewrite rules and mal-rules will be used to detect the error. This will produce a list of different reasons of the error. Figure (5) is the list for the above example. The idea here is to produce feedback for the error from this list.

First proposed explanation	
Removed from Original	2*x+([2]*y)
Second proposed explanation	
Removed from the original expression	
Added to the student's answer	[2*y]+2*x
Third proposed explanation	[y]+2*x
Removed from the original expression	
Added to the student's answer	2*([y]+x),
Fourth proposed explanation	[y]+2*x
Removed from the original expression	
Added to the student's answer	[2]*(x+y)
	y + ([2]*x)

Figure 5 : An example of different feedback produced by the system

This list usually has two different forms of system feedback. The first form is with a one part explanation, i.e. showing what has been missed from or added to the original expression. The second form of explanation is with two parts, one showing what has been missed from the original expression and the second part showing what has been added to the student's answer. The plan is to model a feedback from this list of explanations. So, if the list has an explanation of the first form, then this is believed to provide a sufficient feedback to the student. This can be prepared by

applying the remove-out technique on the explanation's expression and finding out what has been missed or added to the original expression. This will be enough to identify the error. A hint can be provided with the feedback about the name of the algebra rule that has been mis-used by providing a name for each rule within the system. Consider the list of the explanation in figure (5). From the list, we can see that there is an explanation of the first form (remove from the original 2*x+([2]*y)). Applying remove-out on this expression will lead to the expression ([2]*y), which is clearly identifying that the "y" should be multiplied by "2". At the same time we can locate this "2" in the original expression, since we keep track where the "2" came from in the original expression throughout the simplification process.

In the case of a list without the first form of system feedback (one part explanation), the second form of explanation (two parts explanation) has to be used to provide a feedback. This can be done through the following steps;

Step1- The first step in this case is to compare the two expressions (of the system's feedback) to find the matching sub-terms (with the same root operator, multiplication or addition).

Step 2- If any are found, then they will be removed from the two expressions. If necessary repeats the first step.
If the two expressions have the same root operators then we implement step 3 and 4, otherwise step 5.

Step 3- These will be compared again (using the matching algorithm). This can be done by multiplying or adding a rewriting variable to the simpler expression. The value of the variable will give us an indication of what has been missed from the student's answer. However, if this does not work, then we have to carry out the step 4, otherwise we stop at this point.

Step 4- Transforming the form of negative integers, say (-*N*) to another form; (-1)**N*.

Step 5- If in the comparison step, we find all sub-terms match with different operator root, then this can be use to identify that an operator been used wrongly.

For example, if we assume that the list in figure (6.2) did not have the first form of explanation, then in this case we have to choose one of the other system's explanations, for example;

Removed from the original expression: [2*y]+2*x
Added to the student's answer : [y]+2*x

The first step will be comparing the two expressions to find matching sub-terms which in this case "2*x". This sub-term will be removed from the two expressions, leaving the two expressions "2*y" and "y" to be matched. These two expressions can be matched by multiplying the simpler term (in this case "y") with a variable (say *W*), and looking for sub-terms to be matched for cancellation, which in this case "y". This will leave the value of the variable "*W*" to be "2". This we believe can model a sufficient feedback for the student, which in the case and for the above example; "y" needs to be multiply by "2".

5. Evaluation

The experiment took place in one of the local schools at Liverpool. A sample of children from this school participated in the evaluation. The sample involved 28 pupils from Croxteth Community Comprehensive School, their ages varied between twelve to fourteen years, and were judged by their own teacher to be of average ability at mathematics. The majority of the pupils had used a computer before. The “drill and practice” method is used, where the teacher introduces the subject in advance, and the pupils will practise “what they learned” using the system. It was intended to get an indication (through pencil and paper pre- and post- tests and performances on the system) of whether the system is likely to benefit student learning. A sample of the questions used in both the

pre-test and the post-test, are shown in figure (6). These items were arranged on a paper work sheet for the students to answer. The post-test used comparable mathematical tasks. Similar questions were incorporated in the system, for the purpose of practising. The interviews of the pre-, post-tests, and the experiment with the system were conducted during term time and occupied 5 (50 minutes) sessions.

1. Simplify each of the following expressions
   - 3 (x+7)
   - 3 (x-7)
   - -3 (x-7)
   - -3 (-x-7)
   - 2 (x+y+z)
2. Multiply out these expressions
   - (x+3) (x-7)
   - (x+3) (x+7)
   - (x-3)(x+2)
   - (x-3)(x-2)
   - (-x-2)(-y-2)

Figure 6 : A sample of the pre-test questions

The pupils were encouraged to use as many intermediary steps as they chose. Finally, the pupils were asked to reflect on their own experience by responding to a series of questions, using two different methods; questionnaire and interview.

5.2 Methodology

It was decided to split the class into two groups, according to their mathematical abilities, which were decided using the pre-test. The pre-test was given to a whole class of 12-14 years old pupils. This test is consisted of five questions concerned with the mathematical concepts within bracket manipulations. The answers of the pupils test questions were then marked and from the results, two balanced groups were formed.

Before introducing the pre-test, the researcher explained that a computerised system has been developed to help the student understand a selected mathematical concept, and that the test would provide an indication of where pupils might need help, and could give an indication of the benefits of the system on the pupil's learning. The pupils were told that the responses would have no bearing on their school mathematics marks, but they were urged to do their best. The test was administered to twenty-six pupils in the classroom. All the subjects had received some classroom instruction on manipulating brackets in advance.

The procedure for monitoring the system interactions with the students during the computer sessions is described below. These sessions (which were limited to 50 minutes by the school) concluded with a post-test, which covered comparable tasks to the pre-test task. Overall the experiment took six weeks to administer.

5.3 Experimental Issues

The idea is to split the class into two groups, such that each will have a different way of practising what they learn in the traditional lesson. The first group (Traditional Group) will use the traditional practising methods (pencil and paper) with some exercises presented by the teacher, while the second group (Technical Group) will use the computer system to practise what they learn in the traditional lecture. Finally the pencil and paper post-test was presented to the two groups to see which group gained most benefit from their method of practice.
A demonstration of the software necessary to show the subjects how to use the system, and the procedure followed was that described in the first pilot study.

Before starting the system tasks, the researcher told the students that the computer would present a set of questions, and they must use the system to answer them.

- They could use any of the techniques that they have learned in ways they thought best.
- As they worked, the data would be recorded on floppy disks, which had been provided. The results won't effect your school marks.
- At the end, the researcher told the pupils that he thought they would find it interesting. Do your best, start with the first task now.

The researcher was observing the pupils as they were trying to use the system to answer the tasks. If pupils were unable to proceed with a question, and asked for help, the researcher provided them with limited help (similar to the help usually given with the traditional teaching methods).

5.4 Results and discussions

The answers to the pre- and post-test were marked and the data tabulated for each participant. The aim of the pre- and post-test is to examine the children's knowledge of the targeted task. Similarly, the examples on both the traditional teaching methods and the computer based training methods consider the same concepts.

A Mann-Whitney test of the null hypothesis $E_1=E_2$ (where E_1 is the Median for the traditional group and E_2 is the Median for the technical group) has been performed, as shown in figure (7). A non-parametric test has been chosen because there is evidence that the data are skewed (which is shown clearly in figure (8)). The outcome of this test indicates that there is strong evidence against the null hypothesis (W=117, P<0.05), therefore, significant progress can be achieved, when using the system over the traditional method of practising.

By analysing the data of the test and reviewing the results in figure (8), the following can be concluded;

- There is a significant diversity in performance between the two groups, in favour of the students in the technical group.
- A considerable number of tasks were not solved by students that carried out the traditional method of practice. After training with the system, the pupils gave a response to the majority of the questions.

Traditional Group		Technical Group	
Subject Name	**Marks**	**Subject Name**	**Marks**
DA	0	NE	3
EM	0	SU	5
TH	2	LI	1
GA	4	KE	6
LE	4	JO	4
MA	1	LI	5
MI	1	AN	6
NI	1	DE	8
VI	0	TE	3
JO	3	KI	3
TO	3	LY	8
RA	1	GE	4
GO	6	GI	4
Mann-Whitney Confidence Interval and Test E1 = 1.000 E2 = 4.000 W = 117.0 The test is significant at 0.0028 (adjusted for ties)			

Figure 7: The post-test Results of evaluation study I out of 10

The analysis of the results has shown strong evidence of the difference between the two groups as shown in figure (8). The data must be interpreted with due caution since the study involved only limited numbers of students and tasks. However, the researcher has observed that all the pupils were able to operate the system, and seemed to find the experience motivating.

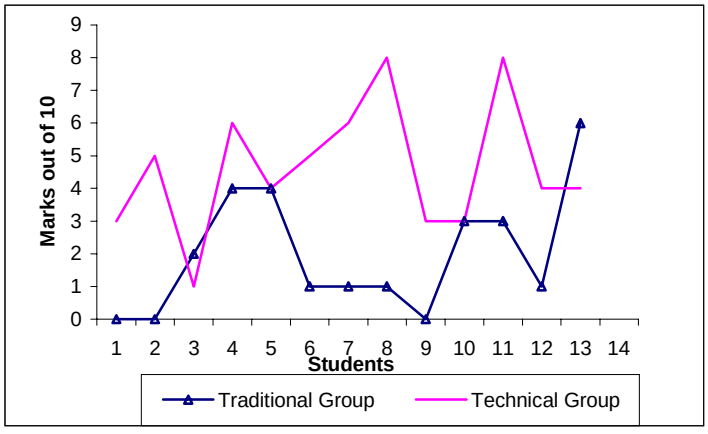

Figure 7: A comparison between the performance of the two groups

Conclusions

There are some basic requirements that need to be considered when carrying out this type of research. Early experimentation is important in order that the system is properly configured to the user requirements. Only an incremental design and implementation process allows for such early feedback. This is especially true when it is necessary to experiment with a prototype to determine learning strategy and usability requirements.

The actual prototype system was tested on several students whose comments and training session data have led to changes being made within the revised prototype system, MathsWeb. The common errors have been analysed and used in the development of the intelligent feedback part of the system. The system can be considered as a success, since the results of the teaching experiments have shown the positive effects of using the system as a training environment (for learning manipulative algebra skills) as opposed to the traditional training method of pencil and paper.

References

Al-Jumeily, D. Intelligent Algorithms and Software for Computer-Aided Assessment in Mathematics. PhD thesis, Liverpool John Moores University, 2000.

Al-Jumeily, D. and P. Strickland (1998). An Intelligent Computer Algebra System for the World Wide Web - MathsWeb. International Conference on the Teaching of Mathematics, Samos, Greece, Wiley.

Al-Jumeily, D., P. Strickland, et al. (1998). Evaluating the Use of an Intelligent Computer Algebra System as a Learning Tool;. Proceeding of ELT 98: Innovation in the Evaluation of Learning Technologies, University of North London.

Anderson, J., C. Boyle, et al. (1990). "Cognitive Modelling and Intelligent Tutoring." Artificial Intelligence **42**(1).

Anderson, J., C. Boyle, et al. (1990). "Cognitive Modelling and Intelligent Tutoring." Artificial Intelligence **42**(February): 7-49.

Anderson, J., C. Boyle, et al. (1985). "Intelligent Tutoring System." Science **228**(6): 456-462.

Anderson, J. R. (1992). Intelligent tutoring and High School Mathematics. ITS'92 Lecture Notes in Computer Science, Springer Verlag.

Beeson, M., Ed. (1989). The User Model in MATHPERT: An Expert System for Learning Mathematics. Artificial Intelligence and Education: Synthesis and Reflection. Amsterdam, I.O.S.

Beeson, M. (1990). "MATHPERT: A Computerised Learning Environment for Algebra, Trigonometry, and Calculus." Journal of Artificial Intelligence in Education **1 (4)**.

Beeson, M., Ed. (1992). Mathpert: Computer Support for Learning Algebra. Logic programming and automated programming, Lecture notes in computer science, Springer-verlag.

Beeson, M., Ed. (1998). Design Principles of Mathpert: Software to Support Education in Algebra and Calculus. Computer-Human Interaction in Symbolic Computation, Springer.

Bundy, A., F. Harmelen, et al. (1990). Extensions to the rippling-out tactic for guiding inductive proofs. 10th International Conference on Automated Deduction,, Springer-Verlag.

Buchberger, B. (1990). "Should Student Learn Integration Rules?" SIGSAM Bulletin **24**(1): 10-17.

Hayden, M. and E. Lamagna (1998). “Newten: An Interactive Environment for Exploring Matheamatics.” Journal of Symbolic Computation **25**: 195-212.

Delozanne, E., E. Bruillard, et al., Eds. (1992). Mathematical Learning Environments: A French Viewpoint. Mathematical Intelligent Learning Environments/Hyacinth, NWANA.

Okamoto, T. (1992). “Student Modelling and Its Diagnosis Based on Synthesis of Sets of Production Rules.” Electronics and Communications in Japan, Part 3 **75**(10): 275-285.