USING INTELLIGENT ALGORITHMS TO GUIDE A BEST SOLUTION EXPLANATION MODEL FOR AN INTELLIGENT TUTORING SYSTEM IN ALGEBRA MANIPULATION

Jordan .C .P KUNG, Paul STRICKLAND, Dhiya AL-JUMEILY, A. TALEB-BENDIAB
School of Computing & Mathematical Sciences
Liverpool John Moores University
Byrom Street, Liverpool, L3 3AF, UK
{cmsjkung, p.m.Strickland, d.aljumeily,A.Talebbendiab} @livjm.ac.uk

ABSTRACT

The aim of *MathWeb* is to augment students' learning processes in algebra manipulation. Previous research has shown that the system can provide more effective learning experience than using traditional methods. In this paper, we describe the architecture of *MathWeb II,* which is based on a set of new algorithms including a model tracing reasoning mechanism and term rewrite technique to support students' manipulation skill in linear equations. One of the most important aspects of the *MathWeb II* is to provide "optimal solution" explanation to inform the student how to solve linear equations in more effective ways which can help the student to have a better understanding about their learning process. The paper starts with a brief description of the *MathWeb II* system's architecture. This will be followed by a detailed presentation of the organisation of the best solution explanation model for linear equations. Finally, the paper will draw some general conclusions and present a description of some further work.

1. Introduction

Many researchers have agreed on the benefits of using such systems to improve the student's learning process in mathematics [9]. However, the student may answer the algebra question correctly but in a tortuous, inefficient and complex manner. As a result, the overall student performance will be reduced due to the time used to consider for processing unnecessary calculations. Experience has shown that students will gain a better understanding of manipulation skills if they are exposed to the considered "ideal" problem solution. This has formed our motivation to develop appropriate support in order to study and model this observation. Therefore it is necessary to teach the student how to calculate the algebra question in a more efficient way.

This paper describes the theory behind the development of an intelligent algebra tutoring system (*MathWeb II*), which can be used to improve the student's learning process in linear equations. An overview of the *MathWeb II* is given containing the functional facilities of the system. The system has been developed for the purpose of examining the student answer step by step and provides "optimal solution" explanation to inform the student how to solve different linear equations in a better way when the student answer is correct but it is not the best solution. In order to provide "optimal solution" explanation, a generative approach is developed based on a set of new algorithms with the previously developed model tracing reasoning mechanism [3] and term rewrite technique [7]. Two types of "optimal solution" explanations are provided for improving the learning process in solving different linear equations. The first type shows the student how to find a best solution step while the second type is to show all the best solution steps for solving the whole equation.

2. The Architecture of MathWeb II

MathWeb II is an intelligent tutoring system for algebra manipulation. The system's logic and operation is based on the already developed term rewrite technique [2], [7] and model tracing reasoning mechanisms [3], [8]. The purpose of the new *MathWeb II* system is to provide the best solution explanation in order to improve the student's algebra manipulation skill with a 'learning by doing' environment. The current system capability is limited to polynomials and linear equations. As shown in figure 1, the system's architecture consists of a set of expanded components. These include a user-interface, a best solution explanation model and a student model including diagnostic [2] and performance model [8].

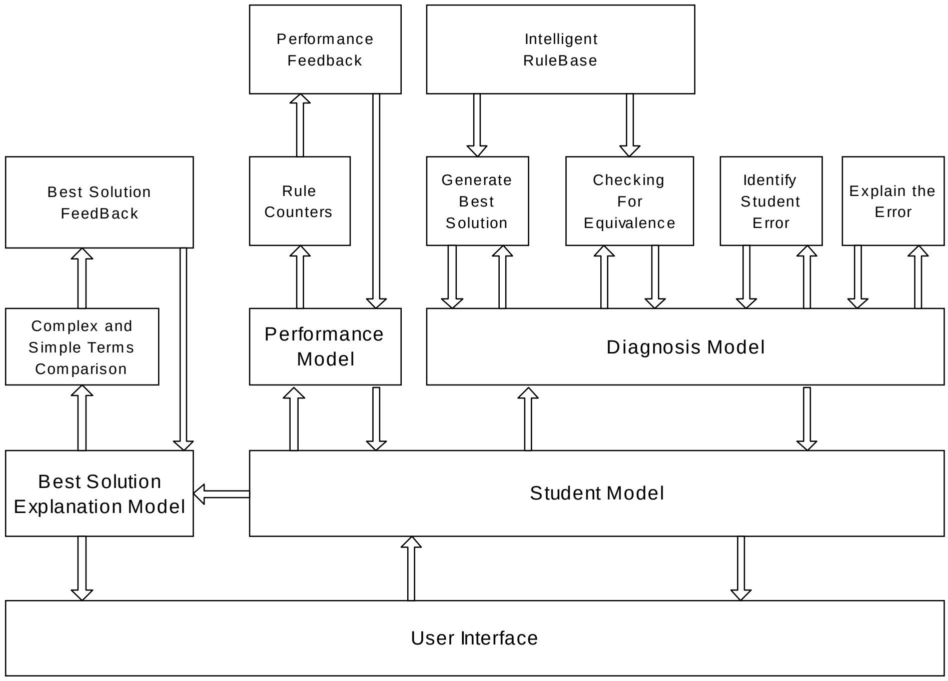

Figure 1: The MathWeb II system configuration

2.1 User Interface

When the answer is inputted by the student, an interface component is required which presents the question to the student, provides an input tool for the student to enter various algebra answers, recognises the student's response, and return the feedback message. The user interface component also has a function to recognise a typing error, which gives a message to urge the student to re-input the expression.

2.2 Student Model

A student model can be separated into two sub-components: diagnosis and performance model [3]. The diagnosis model can be described as the process of getting information concerning the student behaviour. The performance model can be described as a set of data structures to record the data generated by the diagnostic module.

2.2.1 Diagnosis Model

The purpose of the diagnosis model is to analyse the student answer, identify any student error and provide a suitable explanation according to the student response. This can be done by transferring the subject knowledge into many different sets of rewrite rules, which transfer a term (polynomials and linear equations in this case) to another equivalent expression. The rewrite rules include not only the correct rewrite rules, but also include other rewrite rules which are organized into several sets, namely transparent rules, mal-rules, and linear equation rules.

The first set is the correct rewrite rules include two types of rewrite rules, regular rewrite rules, and conditional rewrite rules. 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. The second set of rewrite rules is the set of transparent rules, which can be defined as basic algebra rules that should be well known by the students. The third set of rewrite rules is the linear equation rules, which can be used in the process of solving integer linear equations with one unknown. The final set of rewrite rules is consists of incorrect rewrite rules (mal-rules), which can be used to express the errors that students make [2]. Table 1 presents part of a set of rewrite rules, which can be used to simplify polynomial and linear equation with one unknown to a solved form, specifically for the algebra domain of integer polynomials and linear equation.

Correct Rules	
Rules	**Semantics**
$A+A \longrightarrow 2*A$	*Addition of unknowns*
$A*(B+C) \longrightarrow A*B+A*C$	*The distributive law*

Transparent Rules	
Rules	**Semantics**
$0+A \longrightarrow A$	*Adding zero to unknown*
$+(A) \longrightarrow A$	*Positive sign*

Linear Equation Rules	
Rules	**Semantics**
$(M*X=N) \longrightarrow (X=N / M)$ where $M \neq 0$	*Dividing by the coefficient of the unknown (Isolation)*
$M*X \pm N = 0 \longrightarrow M * X = \mp N$	*Add- subtract of the unknown*

Table 1: Example of rewrite rules

If the student inputs an incorrect answer, then mal-rules can be used to express the errors made. For example, if the problem is to solve a linear equation 4+4*(x-1) = 2 and the student may enter a step 4+4x–1 = 2. After the comparison process using rewriting and evaluation techniques (described in section 4), we know that the student answer 4+4x–1 = 2 is incorrect as it is not equivalent to 4+4x–4 = 2. In order to find out the type of student error, the mal-rule A * B $\longrightarrow$ B is applied to generate an incorrect system answer 4 + 4x + **4*-1** = 2 $\longrightarrow$ 4 + 4x + -1 = 2 which is equivalent to the student answer. Then we know this student may have a problem in using the distributive law.

2.2.2 Performance Model

The idea of the performance model is to use a set of rule counters to store what types of errors are made by the student during exercises. As a result, the system will generate performance feedback. The performance feedback will not only contain the result of student performance, but also a detailed explanation of their performance (why the student made these errors).

In order to provide accurate performance data, the performance model will use rule counters to store the numbers of different rewrite rules used within each step in the student's solution. There are two different types of rule counter, for logical and non-logical errors in the performance model. The rule counters of non-logical errors will also contain a set of sub-counters to identify the incorrect operator used to simplify an integer polynomial or a linear equation. The performance model uses the diagnosis model to identify, as well as to locate the student errors [8]. The following is an example to show how this model analyses the student performance.

Mal-Rule	Modify System Output	Student Answer
A * B -> B	4 + 4 * x + **4** *** -1** = 2	4 + 4x – 1 = 2

Performance Counters	Logical		Non - Logical				
	Left Distributive	Right Distributive	Addition +, -, *, /	Subtraction +, -, *, /	Multiplication +, -, *, /	Division +, -, *, /	Sign
	1	0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0, 0, 0, 0	0

Table 2: Identify student error using mal-rule for left distribute law

From the result of the rule counter, we can identify the status of the student performance clearly. In this case, the performance model will have some information to suggest that the student has a misunderstanding of the left distributive law “A * (B + C) $\rightarrow$ A * B + A * C”.

3. Best Solution Generation

In order to generate the best solution for a particular linear equation, we propose a new algorithm (optimal solving for linear equations) with our developed rewrite rules [2][7] and model tracing reasoning approach [3][8] to generate a best solution with a minimum number of reasonable steps of simplification for the linear equation.

3.1 Polynomial Optimal Tree

The idea of the polynomial optimal tree is to generate a best solution for each step simplification using the minimum number of steps to achieve the final correct answer. This can be done by building a problem solving strategy with the use of a set of rewrite rules [6][7]. The structure of the problem solving strategy is based on a binary tree format. This algorithm divides an equation into sub terms, using the priorities of the operations. For example, to generate a problem solving strategy for expanding a polynomial 2 * (x + 137 – 131), the system will analyse the polynomial structure and then divide it into sub polynomials based on the priority for each operator.

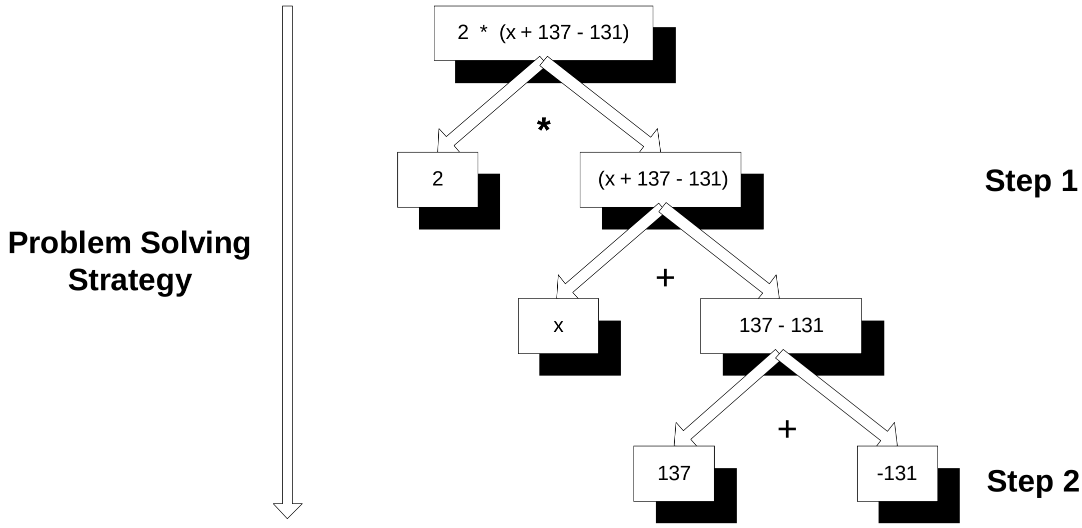

Figure 2: Generate problem solving strategy for a polynomial

From the problem solving strategy, there is an "optimal" solution generated for this particular polynomial. The best solution is represented in a set of ordered steps. Each ordered step is identified as simplifying the smallest sub polynomial. Therefore, we can manipulate any polynomials with rewrite rules in a best way by following these ordered steps.

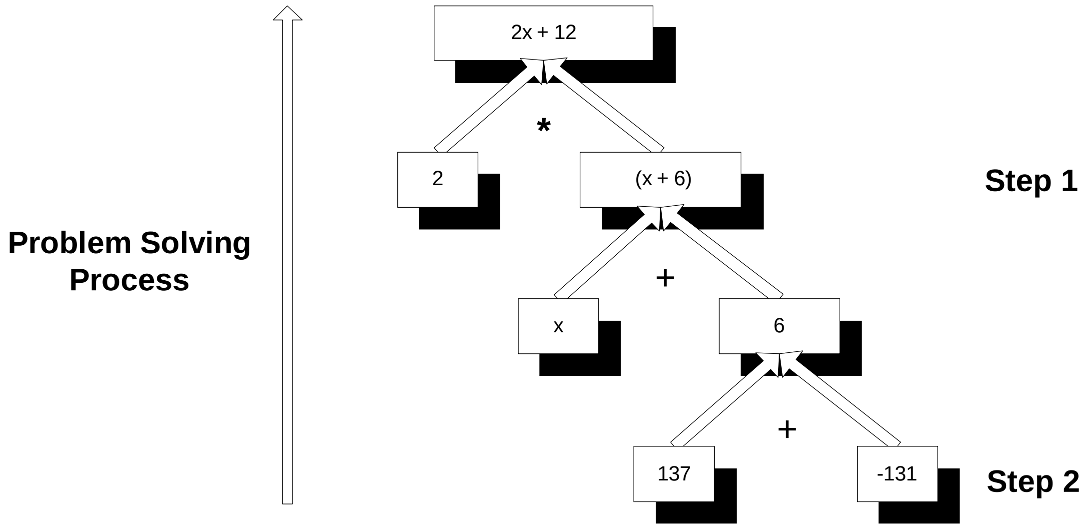

Figure 3: Optimal problem solving process

3.2 Optimal Solving For Linear Equation

The purpose of this algorithm is to provide an "optimal" solving strategy for linear equations, which can be used with our rewrite rules and polynomial optimal tree in order to simplify different linear equations with one unknown in a best way. This algorithm will first generate the problem solving strategies (polynomial optimal tree) for the polynomials on both the left and right hand sides of a linear equation. The polynomials (on both left and right sides) will be simplified based on the problem solving strategy with rewrite rules to obtain the simpler form. After the polynomials are simplified on both the left and right sides, the linear equation rewrite rules are applied to move the unknowns to one side and move the numerical terms on the other side. Finally, it will apply associated calculations to the numerical terms to obtain the final answer for the unknown. This can be done with the following steps.

Step 1: Use polynomial optimal tree algorithm to generate problem solving strategies for the polynomials on both the left and right sides.

Step 2: Apply polynomial rewrite rules to these polynomials to obtain the fully expanded forms.

Step 3: Use linear equation rewrite rules to move a term to another side.

Step 4: Apply polynomial rewrite rules to the modified polynomials. If there is not a final answer for the unknown x then go to step 3 otherwise stop process

For example, to solve a linear equation $x - 4 + 2 = 2 * (3-1)$, it will divide the linear equation into two polynomials $x - 4 + 2$ (the polynomial on left-hand side) and $2 * (3-1)$ (the polynomial on right-hand side). Then it simplifies these polynomials to obtain the expanded forms as x – 2 and 4. After that it moves the unknown to one side to form the equation into the final structure x – 2 = 4 -> x = 4 + 2. Finally calculate the value for the unknown x as 6.

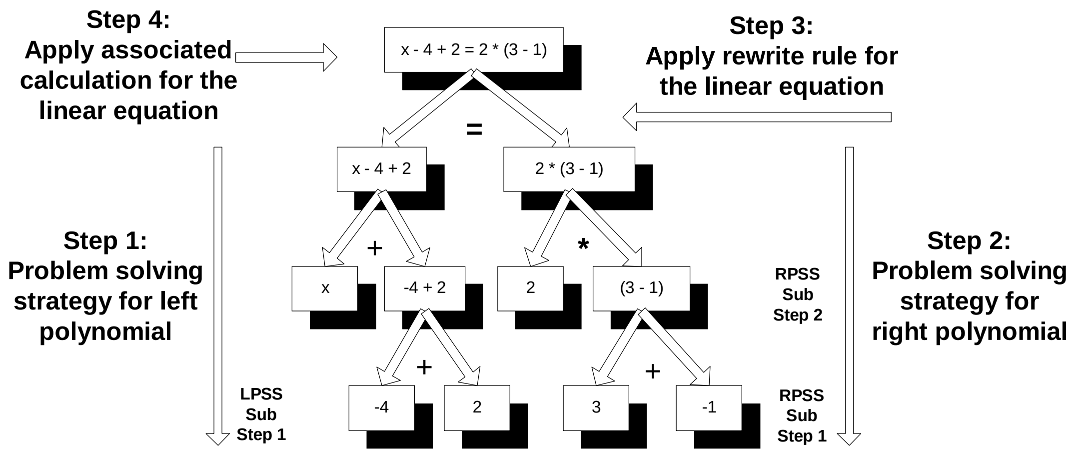

Figure 4: Generate optimal solving strategy for a linear equation

In the above "optimal" solving strategy for the linear equation, it will execute the steps within the Left Problem Solving Strategy (LPSS) first and then process the Right Problem Solving Strategy (RPSS).

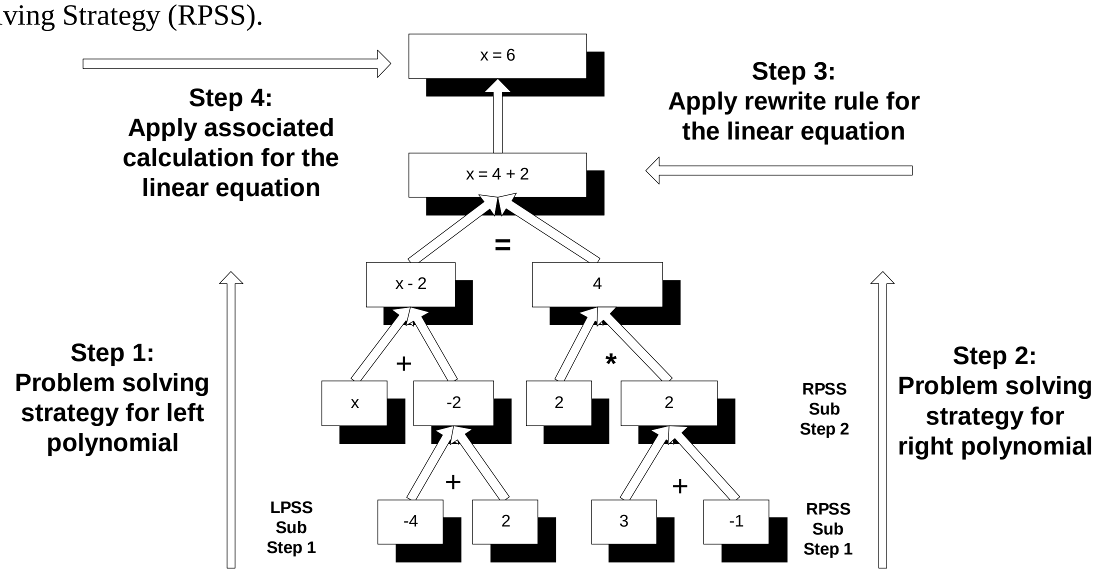

Figure 5: Optimal problem solving process for a linear equation

4. Checking For Equivalence (Rewriting and Evaluation)

We also need to check the correctness of each simplification within each step in the student's solution for a linear equation. This can be done through two stages of validation process as rewrite technique [7] and evaluation [4]. The first stage of the validation process is to apply rewrite rules to the linear equation in order to obtain the value for the unknown. After that, this value is used to evaluate the student answer for checking the equivalence. For example, a question is to solve a linear equation 3x - 4 = 2x + 4 and the next step student's answer is 3x = 2x + 4 - 4. In this case, a set of rewrite is applied to the "optimal" solving for linear equation (described in section 3) to obtain the value for the unknown x as 8.

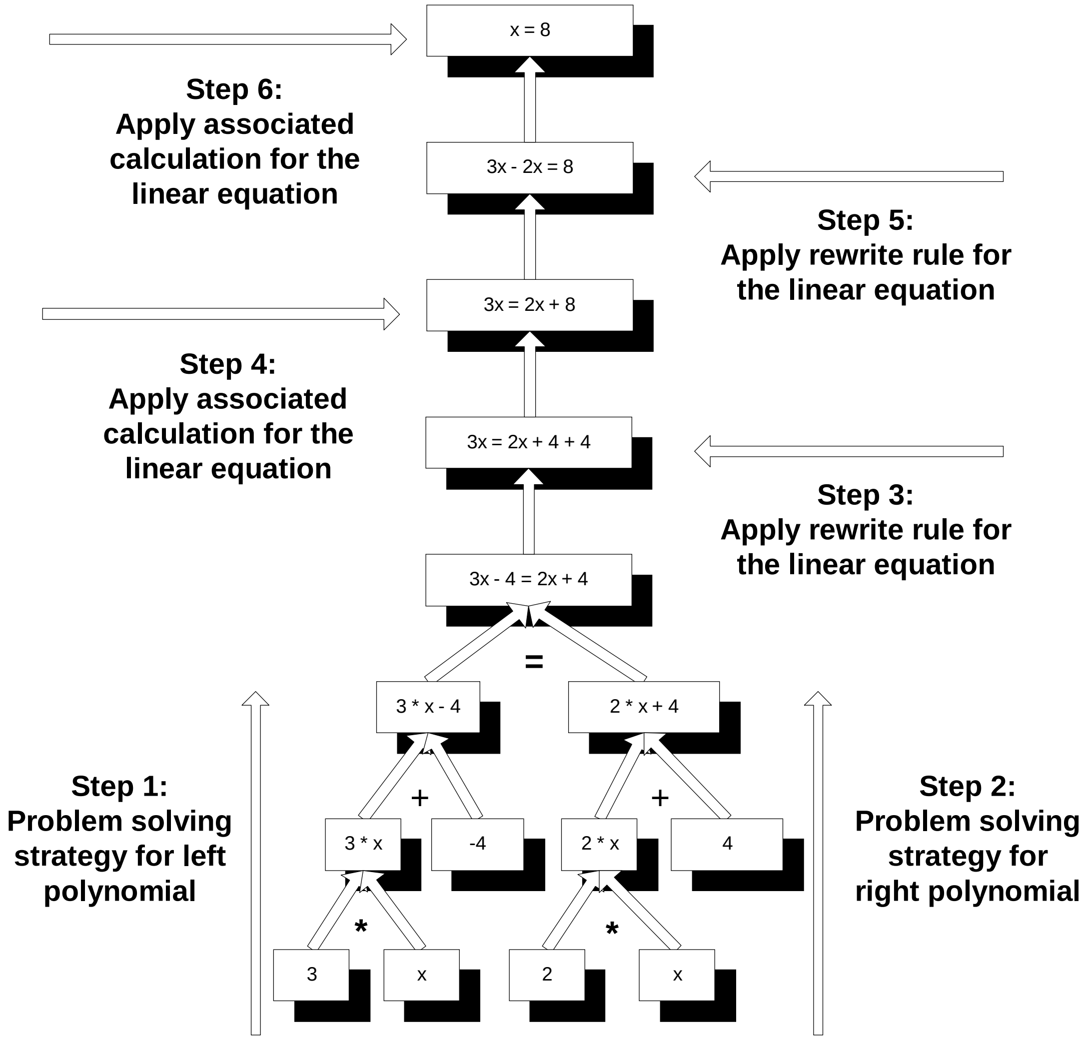

Figure 6: Calculate the value for unknown x with optimal solving strategy for linear equation

After we have obtained the value for the unknown x, the evaluation process is undertaken for checking the equivalence of student and correct answers. This process can be done, by evaluating the student answer with a correct substitution of the unknown, so that the values are examined on both sides of the student's answer. For the above example, the student answer is incorrect because the values are different 3x = 2x + 4 – 4 -> 24 = 16 on both sides after the evaluation.

5. Best Solution Explanation Model

In this section, we purpose a new best solution explanation model to provide the best solution explanation for linear equation. The purpose of best solution explanation model is to improve the student's algebra manipulation skills with a 'learning by doing' environment [5]. We believe to obtain the simpler form for sub polynomial first is always the best way to solve a linear equation. The best solution explanation can be generated based on the "optimal" solving strategy for linear equation described in section 3. For example, if the problem is to simplify a linear equation 2 * (x + 137 – 131) = 4 + 1 –3 and the student decides to simplify the left polynomial 2 * (x + 137 – 131) first. Then the best solution explanation model will check the correct student answer with the problem solving strategy for this polynomial to identify whether it is a best solution or not. If the student input for next step is 2x + 274 – 262, then the best solution explanation model will inform the student that the best solution is to first simplify 137 – 131 before expanding 2 * (x + 6). On the other hand, if the student decides to simplify the right polynomial 4 + 1 – 3 and the student answer for next step is 4 – 2, then the best solution explanation model will inform the student that the best solution is to first calculate 4 + 1 before subtracting 5 - 3. The best solution explanation model will also inform the student with a best solution explanation when the student move a polynomial to another side as 2 * (x + 137 – 131) –3 –1 + 3 = 0 before these polynomials in the simpler form on both side. This can be done through the following steps. Suppose that the previous step was the equation pL(x) = pR(x), and the student enters the step sL(x) = sR(x).

Step 1: Apply polynomial sorting to form both the correct and student answers in the same format in order to identify the student action. If either sL(x) is different than pL(x) or sR(x) is different than pR(x) then go to step 3. If sL(x) and sR(x) are both different than pL(x) and pR(x) then go to step 2.

Step 2: If the both sL(x) and sR(x) are not in the simpler form, then go to step 3. Otherwise stop process as the student answer is a best solution.

Step 3: Calculate the minimum number of steps to achieve the normal for the both correct and identified student answers. If the number of student steps is greater than the number of correct steps, then it is not a best solution step and stop the process, otherwise go to step 4.

Step 4: If the number of student steps is less than the number of correct steps, then it is a best solution step and stop process otherwise go to step 5.

Step 5: If the number of student steps is equal to the number of correct steps, then compare the identified student answer and correct answer. If they have the equivalent structure then it is a best solution step and stop process otherwise it is not a best solution step and stop process.

The best solution explanation model can also generate a best solution explanation to show the "optimal ways" for solving the whole linear equation. For example, to simplify the left polynomial of a linear equation 2 * (x + 137 – 131) = 4 +1 – 3, the first step is to simplify 137 – 131 and the second step is to expand the simplified polynomial 2 * (x + 6) to obtain the simpler form 2x + 12. On the other hand, to simplify the right polynomial of a linear equation 2 * (x + 137 – 131) = 4 +1 – 3, the first step is to calculate 4 + 1 and then the second step is to calculate 5 –3 to obtain the simpler form 2. After the polynomials are in the simpler form on both sides, then the best solution explanation model will inform the student that the next step is to move the value 12 to another side as 2x + 12 = 2 -> 2x = 2 – 12. After that the next "optimal" step is to calculate 2 – 12 and then move 2 to another side as x = -10 / 2. The final step is to calculate –10 / 2 to obtain the final answer –5 for the unknown x as x = -5.

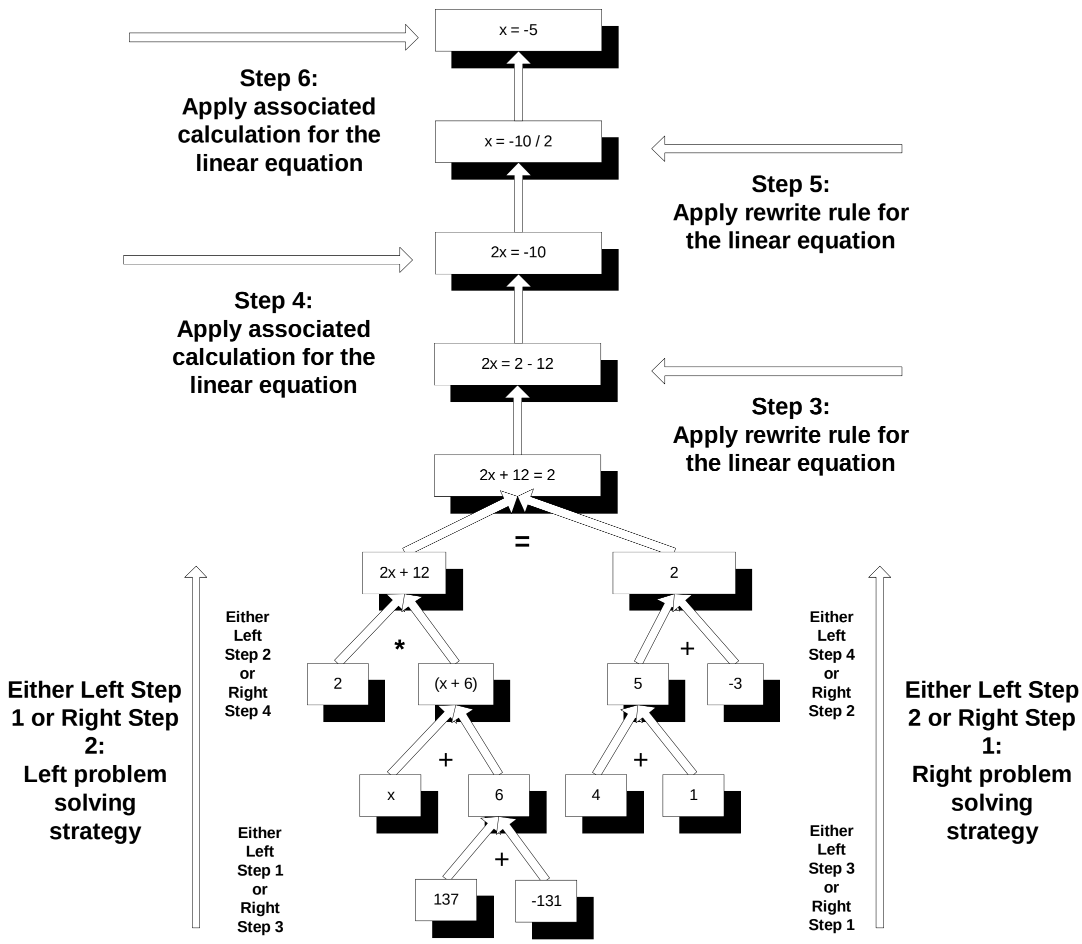

Figure 7: Generate best solution explanation for a linear equation

By informing the student how to simplify the linear equation in an "optimal" way, we believe that the student's manipulation skill can be improved in a better way for manipulating different linear equations.

6. Conclusions

Many researches have agreed on the benefits of using intelligent tutoring systems that it can improve the student's learning process in mathematics. The *MathWeb II* is an intelligent tutoring system, which provides "optimal solution" explanations with a 'learning by doing' environment in order to improve the student's manipulation skill in linear equation. This paper describes the theory behind the development of an intelligent algebra tutoring system (*MathWeb II*). As an overview of the system architecture is given containing the functionality for each model. A set of new generative approaches is also developed to dynamically generate correct answer (optimal solving for linear equation) for different linear equations and provide "optimal solution" explanation in the student's learning process. The idea of the best solution explanation model is to calculate the number of steps to achieve the normal form and analyse the polynomial structure in order to identify whether the correct student answer is a reasonable best solution or not. If the student answer is not a best solution then the best solution explanation model will inform the student how to simplify the next "optimal" step.

The best solution explanation model also provides a best solution explanation for solving the whole linear equation.

As a result of our previous research [1], we believe that the *MathWeb II* can be used to provide more effective learning than doing the same exercise using pencil and paper on your own. However, we still need to prove our system to ensure that the *MathWeb II* will achieve its aim and objective. Therefore, a system implementation is required with a validation study, so that student's manipulation skill can be examined to see the potential effects on student's understanding about their learning process. The experiments will take place in local schools with the student evaluation so that a number of users will use the system to see the affect of using such a system on the students' manipulation skill in linear equation.

References

[1] Al-Jumeily, D. (2001). "An Intelligent Tutoring System for Algebra Manipulation". *The International Journal of Computer Algebra in Mathematics Education*. **8 (3)** 175-194.

[2] Al-Jumeily, D. (2000). Intelligent Algorithms and Software for Computer-Aided Assessment in Mathematics. *Computing and Mathematical Sciences*., PhD thesis, Liverpool John Moores University.

[3] Anderson, J. R., Boyle, C. F., Corbett, A.T., Lewis, M.W. (1990). "Cognitive Modelling and Intelligent Tutoring." *Artificial Intelligence* **42 (1)**: 7 - 49.

[4] Appleby, J. (1997). DIAGNOSYS-A Knowledge-Based Diagnostic Test of Basic Mathematical Skills. *Computer Education*. P. Samuels. **28:** 113.

[5] Burton, R. (1982). *An Investigation of Computer Coaching for Informal Learning Activities. Intelligent Tutoring Systems*., Cambridge, MA: Academic Press.

[6] Dershowitz, N. (1982). "Ordering for Term-Rewriting Systems." *Theoretical Computer Sciences* **17**: 279-301.

[7] Dershowitz, N., Jouannaud, J., Ed. (1990). "Rewrite Systems". Chapter 6 of *Handbook of Theoretical Computer Science: Formal Methods and Semantics*. North-Holland, Amsterdam, J. van Leeuwen.

[8] Kung, J. C. P., Strickland, P., Al-Jumeily, D., A. Taleb-Bendiab (2001). "Using Model Tracing Approach To Guide A Student Model For An Intelligent Computer Algebra System". *14th International Conference of Applications of PROLOG INAP 2001*, The University of Tokyo, Sanjo Conference Hall, Japan.

[9] Polson, M. C., Richardson, J. Jeffrey. (1988). *Foundations of Intelligent Tutoring Systems*. Hove and London, Lawerence Erlbaum Associates Publishers.