Methodologies for the Automation of Web-Based Interactive Courseware for Mathematics

Yukari Shirota[†]

Abstract

In this paper, we describe the newest version of our mathematical e-learning material generation system, e-Math Interaction Agent 2. Its key feature is the simplicity of its solution plans and presentation definitions. The presentation layers are completely separated from the pure mathematical solution processes, significantly increasing the reusability of the definitions.

1. Introduction

In this paper, we describe the newest version of our mathematical e-learning material generation system, e-Math Interaction Agent 2. The system features the separation of the solution and guidance plans. While a solution plan defines the pure mathematical solution procedure for a given set of problems, a guidance plan is the template for the presentation materials, including a virtual character's dialogues. This separation promotes the reusability of both plans. As a result, users (math teachers) do not require Web scripting and programming experience and can easily develop interactive Web-based courseware. They are able to create their own learning materials by defining guidance plans using various visual presentation materials.

In the second section, we explain the history of our e-Math Interaction Agent and the approaches we have taken to meet mathematical lecture requirements. In Section 3, we describe related works and our approach and final goal, pointing out the particular difficulty of implementing interactive courseware. In Section 4, a brief outline of the system architecture is given. The detailed plan definition specifications are explained in Section 5, where the required set of graphs is also identified as our future work. Finally we conclude our paper by describing once again the key feature of our system, the separation of the presentation layer from the solution plans.

2. History and Background

E-Math Interaction Agent 2, the system to be explained in this paper, is the latest and a newly expanded version of our previous e-Math Interaction Agent. We devote much of this section to explaining the history and background of our system, which was developed from April 2003 through December 2005.

†   Faculty of Economics, Gakushuin University, 1-5-1 Mejiro, Toshima-ku, Tokyo 171-8588, Japan.
     E-mail: yukari.shirota@gakushuin.ac.jp.

Although the system is continuously being expanded, there are two main versions. For convenience we call them e-Math Interaction Agent 1 and 2.

The purpose of developing these versions was to solve the following general problems faced in developing Web-based mathematical courseware:

(1) The significant cost of transferring mathematical expression formats to make it possible to display the expressions on Web browsers.

(2) The large amount of time required to properly determine the values of coefficients in math expressions so that students can easily calculate and solve the mathematical problem.

(3) The high cost of creating visual presentation materials, such as graphs and animation, to explain mathematical expressions.

(4) The difficulty of developing interactive courseware for mathematics.

A mathematics teacher must be careful to make effective mathematical courseware and properly define the coefficients in the given mathematical equations as well as the given data and conditions. Particular care is required on the following points:

Confirmation of the existence or uniqueness of a solution for the given mathematical problem.

Verification that the calculation process is not too complicated for the students.

Review of the solution process to ensure that the intermediate solutions are proper as values in the application field.

Teachers must repeatedly rework the courseware materials until they confirm the soundness of the generated learning materials. The cost involved is significant, even if they use mathematical tools. To drastically decrease both the development cost and period, we created our e-Math Interaction Agent 1 in 2003.

We shall explain our approaches to the solution shortly. To solve the first problem above, we used Design Science WebEQ Developers Suite. In our system, the pure mathematical expressions are automatically transferred to Jpeg image files of the expressions using WebEQ, and the names of the automated image files are embedded in the resultant XML file. When a student displays the XML file through a XSLT stylesheet, the automated image files are displayed on the browser.

To solve problems (2) and (3), we used Maple, which makes it possible to solve mathematical problems in symbols. In our methodology for automating mathematical courseware, solution and guidance plans are defined in symbols in advance. When the conditions and data for the given mathematical problem are input, numeric values are automatically calculated by invoking Maple. Graphs are also created by invoking Maple.

The newly expanded function in our e-Math Interaction Agent 2 refers to the methods used to define these solution and guidance plans. A solution plan is the definition of a mathematical (logical) solution plan and describes how to mathematically solve a given problem. A guidance plan is the definition of presentation templates and defines the dialogue spoken by a virtual teacher to a student and how the virtual teacher guides the student using various kinds of presentation material.

In e-Math Interaction Agent 1, we defined the plans for an optimization problem of single variable functions. The solution and guidance plans were integrated and written in Perl. Therefore, the cost of definition was quite high. Moreover, ordinary mathematics teachers lacking Perl scripting capability

could not define the plans. Thus, in the e-Math Interaction Agent 2, we completely separated the solution plans from the guidance plans.

This separation promotes the reusability of the plans. Namely, we can define solution plans and guidance plans separately and store them in plan databases. Then, when necessary, we can select a particular guidance plan from them for the solution plan. The relationship between the solution and guidance plans becomes one to many. Moreover, there may be several solution plans for a given mathematical problem. For example, both Gaussian elimination methods and Cramer's rule can be used to solve simultaneous equations. The teacher who develops the courseware can even select the solution plan. Consequently, the relationship between the solution and guidance plans is many to many.

The most important requirement for this plan definition process is simplicity so that teachers without programming experience can define the plans. The latest version of the definition specifications meets this requirement. These specifications are described later in Section 5.

3. Approaches for Interactivity

In this section, we consider the interactivity of mathematical learning materials. The above-mentioned fourth problem about interactivity is the biggest problem in mathematical courseware development. There are several types of interactivity in mathematics e-learning.

One prime technical problem is how to interactively input and display mathematical elements on Web browsers. The existing input methods were (a) a selection from a list of answers (a choice from among multiple answers) and (b) the input of coefficients into blank sections of equations. Many mathematical input editors are now available to input mathematical expressions. These include commercial products, such as WebEQ editor, a point-click equation editor used to graphically compose equations   in which the input answer equation can be encoded in standard mathematical expression formats, such as MathML.

Various other interactive input methods are also well researched in WWW Interactive Multipurpose Server (WIMS)   . For example, various types of interaction courseware materials are collected as follows: a geometric construction using a ruler and a compass, a plot operation on a plane of complex numbers, Gauss elimination operations for a matrix or set of equations. Yasskin has also developed a set of Web-based courseware with various input methods, "Maplets for Calculus"   . His learning materials are written in Maplet, which is a Maple script language. Commercial product "Maple T.A." is a Web-based testing and assessment system with the computer algebra system Maple as its background system   . Maple T.A. includes an equation editor, an assignment editor, and a question bank editor. These input tools and techniques make it possible to solve most existing interactive input problems and can thereby release users from the limitations of Web-based learning in mathematics. Thus, in the current situation, we could say that the input and display problems of mathematical equations have been solved, at least technically.

Design Science: WebEQ editor, http://www.dessci.com/en/products/webeq/authoring/editor.htm.

WWW intereaction multipurpose server WIMS: http://wims.unice.fr/wims/.

Philip B. Yasskin: Maplets for Calculus, http://calclab.math.tamu.edu/maple/maplets/Contents.html.

Maplesoft: Maple T.A. 2.5, http://www.maplesoft.com/products/mapleta/.

The remaining problem of the equation input concerns how to check the syntactical equality of the input equation and the real answer. For example, consider *(x+y)/2* and *(x/2 + y/2)*. When a student is allowed to input an equation in a free form through an equation editor or in LaTex or similar formats, such as *x^2+sqrt(y*z)/2*, the student can input various forms of the answer even if the input is the correct answer. Goguadze defines three types of equalities: 'numeric equality', the above mentioned 'syntactic equality', and 'semantic equality' which means whether the student's input is the same mathematical object or not .

Symbolic calculation functions or computer algebra systems are necessary to check syntactic equality. For example, Maplesoft advertises it as the feature of Maple T.A. However, if the answer input by the student is not sufficiently simplified, how should we ask the student to simplify or normalize the equation and in what form? This remains an issue. In other words, a heuristic human teacher's activity and some feedback implementation is required. This, specifically, is our future research destination. At any rate, the problem of syntactic quality among input could also be solved by developing sophisticated equality checking programs with symbolic calculation functions.

Next let us consider interactive exercises. Much research has been done on the generation of interactive exercises . Such exercises consist of a finite state machine of interactions, each interaction containing feedback for the event trigger; an interactivity assignment that describes how to substitute certain parts of the feedback by interactive elements; and transition maps describing which interactions come next, depending on the user's input . The research point is how to describe simply the transition map. Many languages and description tools have been proposed: IMS QTI (Question & Test Interoperability) , AIM (Assessment in Mathematics) , Maple T.A., and WaLLis .

The target of such research on interactive exercises is to describe a mathematical drill that a student will solve. Our research target, however, is a mathematical word problem, not a drill or an exercise. We would like to develop learning materials that explain and guide mathematical algorithms and procedures. Our final goal is to simulate the cleverest and most skillful teacher's interactive guidance on a computer system. Such an interactive guidance function has not yet been implemented. The works of famous mathematician, Pólya, offer much inspiration for solving mathematical problems[10,11]. We would like to implement his heuristic strategies. Melis and Cairns are trying to implement his strategy in the systems[12,13] of their ActiveMath project, and we could say that our research direction is similar to their re-

G. Goguadze, A. González Palomo, and E. Melis: "Interactivity of Exercises in ActiveMath", International Conference on Computers in Education (ICCE 2005), http://www.activemath.org/publications/bibitem.php?index=81.

Manolis Mavrikis, and Alberto González Palomo: "Mathematical, Interactive Exercise Generation from Static Documents", Electronic Notes in Theoretical Computer Science, Vol. 93, Elsevier, pp. 183-201, 2004.

IMS Global Learning Consortium, Inc.: IMS Question & Test Interoperability, http://www.imsproject.org/.

AiM team: AIM, http://maths.york.ac.uk/moodle/aiminfo/.

Manolis Mavrikis and Antony Maciocia: "Incorporating assessment into an Interactive Learning Environment for mathematics," http://ltsn.mathstore.ac.uk/articles/maths-caa-series/june2003/index.shtml.

(demonstration: http://www.maths.ed.ac.uk/~wallis/).

10   George Pólya, How to Solve It, Penguin Books, 1990.

11   George Pólya , Mathematics and Plausible Reasoning, Princeton University Press, 1968.

12   Erica Melis, and Carsten Ullrich: "How to Teach it -- Pólya -Inspired Scenarios in ActiveMath", AI in Education AIED-2003.

search goal.

There are four stages in Pólya's problem solving process: (A) understand the problem, (B) devise a plan, (C) carry out the plan, and (D) examine. Pólya stated that the teacher must explain and ask students proper questions to unobtrusively help them understand the problem and devise a plan, which may occur suddenly after repeated unsuccessful trials, as a bright idea. In Pólya's strategy, the teacher will ask students questions, such as: "Do you find the connection between the data and the unknown?" "Do you know a related problem?" "Do you know a theorem that could be useful?" "Here is a previously solved problem related to yours. Could you use it?"

Our planning approach is to store theorems, mathematical problems, and solution plans in a database. The metadata are added on the stored data. The metadata are used to find the relationships among the data. This approach may be related to Melis' interactive concept mapping[14]. With our final goal in mind, we have developed e-Math Interaction Agent 2, in which the following basic functions for courseware generation have been implemented:

(1) Conversation functions by a virtual character.

(2) Metadata description on the conversation templates by which the system can replace the metadata description parts with concrete variable names or equations.

(3) Creation of various presentation materials.

In the later sections, these functions will be explained through the automation of sample learning material.

4. System Architecture

In this section, the system architecture of e-Math Interaction Agent 2 will be briefly explained (See Figure 1). A detailed explanation can be found in previous papers[15,16].

There are two kinds of process in the flow of the automation. The first is to define the solution/guidance plans for the problem type. So far we have defined the two problem types: (a) optimization of single variable functions and (b) solving simultaneous equations. In our approach, we define "concepts" as important concepts that appear in the given mathematical word problem. The concept is the metadata and is defined for the problem type to express the contents of the mathematical word problems. Thus, a concept has mathematical/economical semantics.

13   Paul Cairns, Jeremy Gow, and Peter Collins: "On Dynamically Presenting A Topology Course", Annals of Mathematics and Artificial Intelligence, Vol. 38, pp. 91-104, 2003.

14   Erica Melis, Philipp Kaerger, and Martin Homik: "Interactive Concept Mapping in ActiveMath", Proc. of DeLFI 2005, pp 247-258, 2005.

15   Yukari Shirota, "Automatic Generation of Web-based Mathematical Courseware for Solving Simultaneous Equations", Proc. of the 3rd International Human.Society@Internet Conference (HIS) 2005, S. Shimojo et al. (Eds.), Springer Lecture Notes in Computer Science (3597), July 27 - 29, 2005, Tokyo, Japan, pp. 248 - 259, 2005.

16   Yukari Shirota: "Design of Automation Systems for Web-Based Courseware using XML and XSLT", Proc. of Workshop on Databases in Networked Information Systems: 4th International Workshop, DNIS 2005, Springer Lecture Notes in Computer Science (3433), University of Aizu, Japan, March 28 - 30, 2005, pp. 186-203.
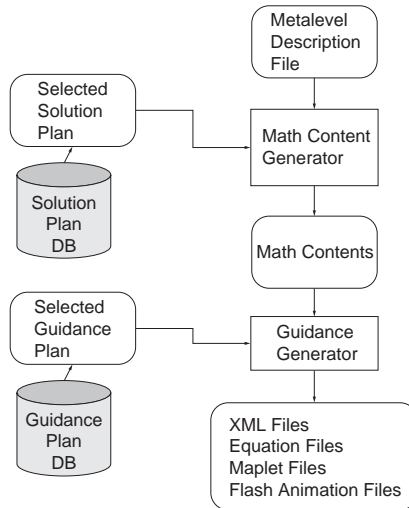
Figure 1: The system flow of e-Math Interaction Agent 2.

A concept has the following five attributes: (1) identification of the concept, (2) the variable symbol of the concept, (3) the equation of the concept, (4) the condition on the concept variable, and (5) the value of the concept variable. These are represented individually as the extensions "id", "var", "eq", "cond", and "value" following the concept name and a period ".".

The second process is to describe data to define a target mathematical word problem. For example, the word problem may be a national income determination problem. The definition data called "metalevel description data" for this example are shown in Figure 2. In the last line of the metalevel description data, the problem type name "equilibrium" and the two parameters "Y" and "Ys=Yd" are illustrated. The problem type has been defined to solve the given set of simultaneous equations to find the equilibrium value of the unknown.
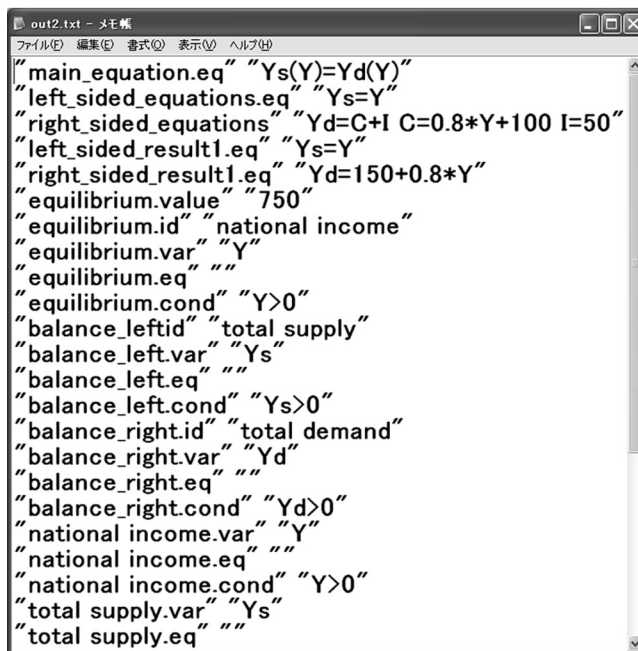
data: national income, Y, ,Y>0
data: total supply, Ys, ,Ys>0
data: total demand, Yd, ,Yd>0
data: consumption, C, C=0.8*Y+100, C>0
data: investment, I, I=50, I>0
unknown: national income
given: consumption, investment
relationship: Ys=Y;Yd=C+I
find: equilibrium(Y, Ys=Yd)

Figure 2: Metalevel description file to define this
national income determination problem.

Following the input of the metalevel description data, the math content generator generates the math contents. The math content generator solves the problem following the solution plan defined in advance. In this stage, the system solves the given problem using symbolic calculation functions.

In this case, the solution plan "equilibrium" is called up. The concepts defined include "equilibrium" (the unknown) and "main equation" (the main equilibrium equation). Figure 3 shows the generated math content. In Figure 3, the calculated equilibrium value 750 and the solved equation of *Yd(Y)* can be seen. The given data defined in the metalevel description file are also added there as the concepts. The math contents are pure mathematical contents without any presentation materials or elements.

Figure 3: The generated math contents.

Next, the system creates the target learning material with the presentation material as an XML file. One guidance plan corresponding to the problem type is selected and input into the guidance generator. The guidance generator, using the input math contents generated in the previous stage, substitutes the guidance plan (template) to create XML files. One of the generated guidance XML files, which corresponds to the upper window portion of the screen, is shown in Figure 4. Finally the student can see the XML file through the XSLT stylesheet, as shown in Figure 5.

Figure 4: Generated XML file to be displayed as learning materials.

Figure 5: The screen image of the generated learning materials.

5. Plan Definition

In the section, we shall explain the definition of plans and guidance.

5.1. Solution Plan Definition

Teachers who will develop learning materials on e-Math Interaction Agent 2 have to define the solution plan of the problem type in advance. Alternatively, they will be able to use the solution plans previously defined by other teachers. In the definition process, the teacher first defines concepts to be used and describes the solution plan using these concepts. For example, we defined the following concepts for the problem type "equilibrium": (1) the unknown variable, (2) the left part of the main equilibrium equation, and (3) the right part of the main equilibrium equation.

The definition is described as illustrated in Figure 6. The symbol "<=" means an assignment of the returned value. As shown here, only assignment statements exist in the definition statements: there is no control mechanism, such as "if then else" statements and "while" loops. Such controls are involved in the invoked functions, such as "solveeqs()," to be invisible to the users. The invoked functions are written in Open Maple API. Then the necessary symbolic calculations are executed by Maple.

```
equilibrium(equilibrium.var, main_equation.eq)
balance_left.var <= left(main_equation.eq)
balance_right.var <= right(main_equation.eq)
left_sided_equations1.eq <= getRelations(all_eqs.eq, balance_left.var)
right_sided_equations1.eq <= getRelations(all_eqs.eq, balance_right.var)
left_sided_result1.eq <= solveeqs(left_sided_equations1.eq, balance_left.var, equilibrium.var)
```

Figure 6: A definition of the solution plan for "equilibrium."

5.2. Guidance Plan Definition

Next we shall explain the guidance plan definition. The guidance plan is defined using the concepts and XML tags. The XML tags are defined in advance for the presentation stylesheet. In Figure 7, a sample guidance plan definition is shown. There the underlined description parts correspond to the defined concepts (metadata). Real values (or equations) are substituted for these concept parts. After the substitutions, the functions to create presentation materials are invoked. For example, the function makeGraph is invoked to make a two-dimensional graph of the given equation.

The tags "<stat>" and "<equa>" individually mean a normal statement and a mathematical equation; the equation part will be automatically transformed to be the image file of the equation. As shown here, the virtual character's conversations can also be described in concepts. This promotes the reusability of the guidance plan, because it maintains independence from the given definition of an individual word problem.

Now we shall discuss the kinds of graphical object required as a part of the generated presentation materials. From our experience, at a minimum the following graphical objects are essential.

（1）Mathematical two-dimensional graphs.

The stationary points (max, min, or inflection) should be marked on the graph. The point of this graph

<title><id>Prob.1</id><stat>Income Determination </stat></title>
<statement>
<stat>The commodity market is in equilibrium when the </stat>
<stat>supply, total supply.var, matches the demand, total demand.var.</stat>
<url>makeGraph(left_sided_result1.eq)</url>
<url>makeGraph(right_sided_result1.eq)</url>
<stat> Assume a simple two-sector economic model in which </stat>
<equa>Yd=C+I.</equa>
<stat> Find the equilibrium level of income, national income.var, </stat>
<stat>if the consumption function is </stat>
<equa>consumption.eq</equa>
<stat> and autonomous investment, </stat>
<equa>investment.eq.</equa>
<url>makeCrossGraph(left_sided_result1.eq, right_sided_result1.eq)</url>
</statement>

--
<guid>First, draw the graph of balance_left.id.</guid>
<guid>Then, the graph of balance_right.id.</guid>
<Q>What is the equation of balance_right.id?</Q>
<A>right_sided_result1.eq.</A>
<guid>Finally, find the crosspoint of the two graphs.</guid>
<Q>What is the final answer for equilibrium.id?</Q>
<A>equilibrium.value</A>

Figure 7: The sample guidance plan to create the learning materials shown in Figure 5.

drawing is the interval on each axis. The student would like to watch his/her interested parts of the graph. However, in general, it is difficult to automatically infer their interested intervals of the graph. The following two specialized two-dimensional graphs are also required:

a two-dimensional graph to show the cross point.

a two-dimensional graph to show the effect of a function shift.

Function shifts appear frequently in economical models. When the original function shifts owing to the effect of another variable, the solution position also shifts.

(2) Mathematical three-dimensional graphs.

On the graph, marks are required to show stationary points (maxima, minima, or saddle points). The following two specialized graphs are also required:

three-dimensional graphs with contour lines of z values and corresponding two-dimensional contour maps.

three-dimensional graphs with cross sections by values of y.

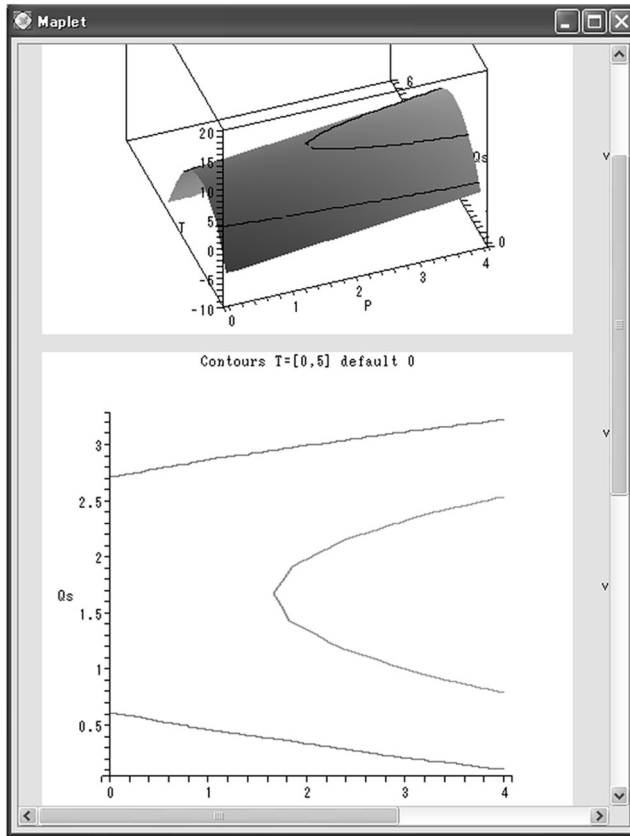The sample images of the graphs are shown in Figure 8 and 9.

Figure 8: A sample of a three-dimensional graph with contour lines
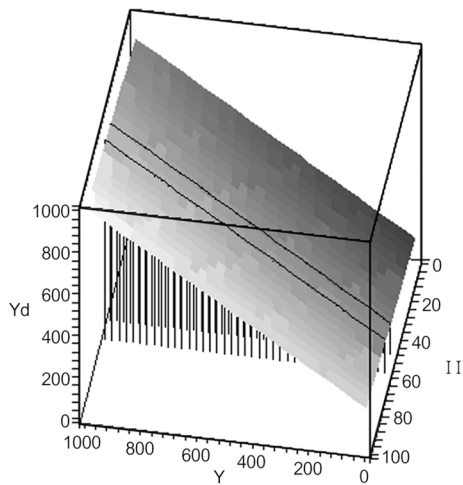and the corresponding two-dimensional contour map.

Figure 9: A sample of a three-dimensional graph with cross sections by values of y.

(3) Schema diagram to show the relationships between data.

To show the relationships between variables among the given simultaneous equations, a schema diagram that looks like an E-R diagram is required. For example, it is necessary that a diagram in which all given variables can be automatically laid out in an attractive form be generated, as shown in Figure 10.
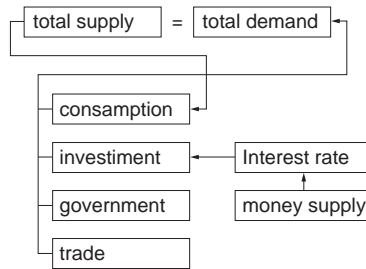

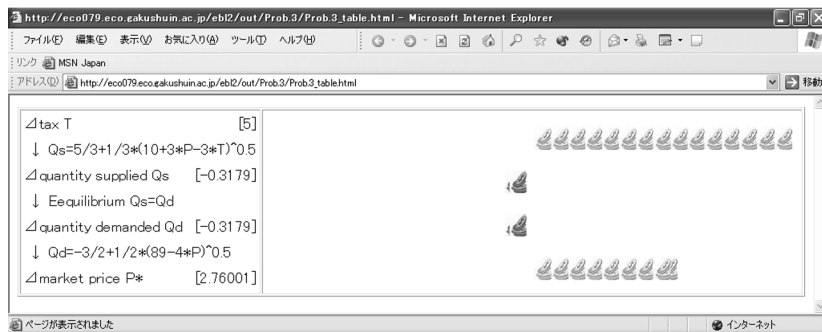Figure 10: A sample image of a relation diagram.

Figure 11: Sample differentiation graphs.

In addition, a differentiation graph is also required so that the effect of one variable variance can be illustrated from the given relationships between the data. The system can calculate the differential values using the partial differentiation facilities of the mathematical software. In e-Math Interaction Agent 2, this graph function has been implemented as Flash animation.

When we make learning materials concerning a set of simultaneous equations, the set of functions for graphs described here would meet most users' requirements. While these graph functions have been partly implemented in our system, some further developments are required.

6. Conclusions

In this paper, we have described our e-Math Interaction Agent 2. The system features the separation of solution and guidance plans. A solution plan defines the pure mathematical solution procedure for the

given set of problems. On the other hand, a guidance plan is the template of the presentation materials, including a virtual character's dialogues. This separation promotes the reusability of both plans. Users can define the guidance plan using various visual presentation materials to create their own learning materials.

Concerning the relationship between solution and guidance plans, many guidance plans can correspond to a single solution plan. However, as several solution plans may exist for each math word problem, the relationship can be many to many.

Our goal in developing e-Math Interaction Agent 2 was to enable users to create their own learning materials with various presentations/expressions by using their preferred solution plans. Thus, the research point was the simplicity of the plan definitions. In the latest version of e-Math Interaction Agent, the definition process has been made much simpler than in the previous version. As our future work, we intend to further enrich the function libraries of symbolic calculation functions for solution plans and the function library of visual materials for guidance plans.

Acknowledgement