Automatic Generating Appropriate Exercises Based on Dynamic Evaluating both Students' and Questions' Levels

Akira Suganuma Tsunenori Mine Takayoshi Shoudai

Graduate School of Information Science and Electrical Engineering,
Kyushu University, Kasuga 816–8580, Japan
e-mail: {suga@is, mine@is, shoudai@i}.kyushu-u.ac.jp

Abstract: Popularization of computers and the Internet enable people to hold lectures using Web contents as a teaching material. Although teachers have prepared a lot of Web contents, most of them are used so as only to be browsed by students. If we arrange some exercises according to lecture notes and prepare an answering mechanism for the exercises via the Internet, students can attempt the exercises any time. This paper proposes AEGIS (Automatic Exercise Generator based on the Intelligence of Students) that generates exercises of various difficulty levels according to each student's achievement level, marks his/her answers and returns them to him/her. It is necessary for AEGIS to evaluate dynamically both the levels in order that it selects a suitable question for each student. This paper also declares the validity of the method with a simulator.

1 Introduction

As the Internet has come into wide use, WWW environments provide lots of opportunities to various fields. In the educational domain, many Internet technologies enable people to hold lectures using Web contents as a teaching material and even develop new lecture methods using the technologies. Web data are, therefore, being expanded rapidly as useful materials.

We have devoted ourself to develop a Web-based self-teaching system and to build the tools for helping students understand their subjects [Sato 1997, Mine 1998, Fujimoto 1999, Suganuma 2000]. Through our experiences teaching in classes and developing such systems, we recognize the necessity of both a method evaluating students' achievement levels and generating exercises suitable for the students automatically.

For example, in trying to make some exercises for the students in a class, we have to take at least their achievement level into considerations. The well-considered exercises are useful not only to measure the achievement level of the students, but also to improve their performance. Unfortunately, it is not an easy task for any teacher to make exercises with the difficulties suited to their achievement levels. Besides, it is very important to mark their answers to the exercises and return the marked results to them for keeping their learning enthusiasm. These tasks become harder in proportion to the number of the students in a class[Hirokawa 1996].

In this paper, we present the automatic student's achievement level evaluator that generates exercises from tagged documents, presents them to students and marks their answers automatically. We call the system AEGIS (Automatic Exercise Generator based on the Intelligence of Students)[Shoudai 2000, Mine 2000]. AEGIS generates the three question-types from the same tagged data. Guessing the achievement level of each student from his/her trial history, AEGIS selects the most suitable question-type and exercise for him/her according to not only his/her achievement level but also the difficulty level of the exercises. It is necessary for AEGIS to evaluate dynamically both the levels in order that it gives each student a suitable question. Although many CAI systems have been proposed, our system is different from them in the points of re-usability of pre-existing electronic materials and re-estimation of both the levels. We have already proposed the method to re-estimate them[Mine 2000] but have not yet validated it competently. This paper declares the validity with a simulator.

In what follows below, the remainder of this section discusses related works. Section 2 describes the exercise generating process by AEGIS and the specification of the tags designed to generate an exercise. Section 3 describes the algorithm to re-estimate the achievement level of a student and the difficulty level of a question. Section 4 discusses experiments and their results to show the validity of the algorithm described in Section 3, and Section 5 shows the overview of AEGIS. Finally, concluding remarks are given in section 6.

1. **multiple-choice question**
   Complete the sentence. Choose your answer from the following list.
   Data structures need to be studied _____ order to understand the algorithms.
   (1) an (2) in (3) on (4) at (5) by
2. **fill-the-gap question**
   Fill in the blank with the right word.
   Data structures need to be studied _____ order to understand the algorithms.
3. **error-correcting question**
   Right or wrong? Correct the sentence if it is wrong.
   Data structures need to be studied an order to understand the algorithms.

Figure 1: Examples of three question-types

⟨QUESTION SUBJECT=”idioms”⟩
Data structures need to be studied
⟨DEL CAND=”an,on,at,by”⟩ in ⟨/DEL⟩
order to understand the algorithms.
⟨/QUESTION⟩

Figure 2: The tagged data to generate the examples of three question-types shown in Fig. 1

Related Works A lot of automatic quiz generators have been proposed. Browning et al. proposed Tutorial Markup Language(TML for short) to generate questions automatically[Browning 1997, Browning 1998]. TML has a couple of tags to specify a question, a multiple-choice and a message. It requires a correct answer in a multiple-choice tag to mark a student's answer to the question. Carbone et al. proposed CADAL Quiz[Carbone 1997], which generates a multiple-choice quiz from a question database. After marking a student's answer, CADAL Quiz returns the result to him/her and to tutors. Both of them restrict the question type only to a multiple-choice quiz. On the other hand, ClassBuilder[ClassBuilder] generates many kinds of quizzes and grades a student's answer. However, all of them do not mention any effect of making the difficulty level of question-type change according to the students' achievement level. In order to improve their performance and keep their enthusiasm to attempt the quiz for a long time, it is indispensable to consider their performance level for generating their exercise. This point is the difference from other systems. AEGIS makes use of pre-existing electronic documents so as to embed tags in them, generates exercises automatically with tagged documents according to students' achievement levels, and re-estimates both their levels and the difficulty level of the generated question through marking their answers.

2 Automatic Exercise Generating

There can be several types of a question in every subject. Since our aim is to make a computer generate an exercise and mark a student's answer to it, we thus restrict the question-types to the following three: *multiple-choice question*, *fill-the-gap question*, and *error-correcting question*. Figure 1 shows examples of these question-types. All of these question-types can be constructed from a sentence by replacing one or more consecutive words with a blank or a wrong expression. We call the region to be replaced, *hidden region*. We note that these three question-types have different difficulties even though they are constructed from the same *hidden region*.

The exercise generating process from teaching documents is summarized as follows: (1) Setting a *hidden region*, (2) Selecting a paragraph or sentence(s) from teaching documents, and (3) Constructing a candidate list. These three steps are deeply related to the teachers' intentions. It is not easy to extract such intentions automatically from the teaching documents. AEGIS system thus deals with tagged documents including the information such as *hidden region*s and candidate lists.

In order to embed the above three kinds of information in teaching documents, we define the following three tags: QUESTION tag that surrounds a *question region*, DEL tag that indicates a *hidden region*, and LABEL tag that surrounds the relevant sentence/s to the DEL tag. Figure 2 shows the tagged data to be used for generating the examples in Fig. 1. Replacing the word “in” which is located between DEL tags with a blank generates the example of the fill-the-gap question. In addition, the value of CAND “an,on,at,by” constructs the candidate list of the multiple-choice question. On the other hand, replacing the *hidden region* with an element in the value of the CAND generates the error-correcting question. Thus we can generate three question-types from only one *question region* where the above tags are embedded. The additional three attributes of DEL, which contain the information of the difficulty to solve an exercise, are LEVEL, GROUP, and REF. They specify the difficulty of each *hidden*

⟨QUESTION SUBJECT=”W_S”⟩ *question region* ⟨/QUESTION⟩		
W_S	::=	word or symbol, where a backslash (\) must be added just before the symbol if it is a comma(,), double quotes(”), or a backslash(\).
⟨DEL CAND=”CANDIDATE” LEVEL=”PAIR” GROUP=”ID” REF=”ID”⟩ *hidden region* ⟨/DEL⟩		
CANDIDATE	::=	W_S | W_S ’,’ CANDIDATE
W_S	::=	word or symbol, where a backslash (\) must be added just before the symbol if it is a comma (,), double quotes (”), or a backslash (\).
PAIR	::=	LOW ’ ’ HIGH
LOW	::=	an integer between 1 and 10
HIGH	::=	an integer between 1 and 10
ID	::=	keyword
⟨LABEL NAME=”ID”⟩ sentences ⟨/LABEL⟩		
ID	::=	keyword

Figure 3: Definition of tags for exercise generation

region, and the connections to other *hidden region*.

Figure 3 shows the definition of the tags described above in BNF. The LEVEL attribute must be provided because AEGIS initializes the difficulty level of the *hidden region* with its value. On the other hand, the others are not essential attributes. When a value of CAND attribute is utilized as a distractor, AEGIS generates only a fill-the-gap question.

3 Re-estimation of Achievement Level and Difficulty Level

It is very important for AEGIS to estimate an achievement level of a student because it generates exercises according to the level. Such a student level fluctuates constantly because AEGIS measures it whenever the student answers a question. The achievement level of student i at time t is calculated with the following formula:

$$s_{i,t} = \begin{cases} s_{i,t-1} + \dfrac{\sum\limits_{j \in Q} (q_{j,t} - s_{i,t-1})\delta_{i,j}}{\sum\limits_{j \in Q} \delta_{i,j}} & \text{if } \sum\limits_{j \in Q} \delta_{i,j} \neq 0 \\ s_{i,t-1} & \text{otherwise} \end{cases}$$

where Q is a set of questions that he/she answered in the recent 30 trials and $q_{j,t}$ stands for the difficulty level of question j at the time when the achievement level $s_{i,t}$ is calculated. The weight value, $\delta_{i,j}$, is 1 if student i correctly answered question j whose difficulty level is more than his/her achievement level $s_{i,t-1}$ or he/she incorrectly answered question j whose level is less than $s_{i,t-1}$, or 0 otherwise. The achievement level of student i is initialized to 1 when he/she tries a question at first time. The student who has continuously answered a lot of questions correctly, obtains a higher achievement level, and can attempt more difficult one. He/she can try more difficult ones if he/she answers them correctly again. On the other hand, the achievement level of the student who answered most of the questions wrongly, becomes lower, he/she comes to attempt easier ones. Continuously answering them correctly, he/she can gradually attempt more difficult questions.

The difficulty level of a question is carefully configured, because AEGIS uses it to calculate the students' achievement level and refers it to generate a suitable question for a student. Since the teachers set it up with the attribute LEVEL of DEL tag, they can assign the upper and lower limits of the difficulty level of a *hidden region*. However, there may be a gap between the difficulty level evaluated by the teachers and that found by the students. A question evaluated by the teachers as an easy one, may not always be answered correctly by lots of students, and vice versa. AEGIS, therefore, utilizes the value of the attribute LEVEL as an initial value of the difficulty level of

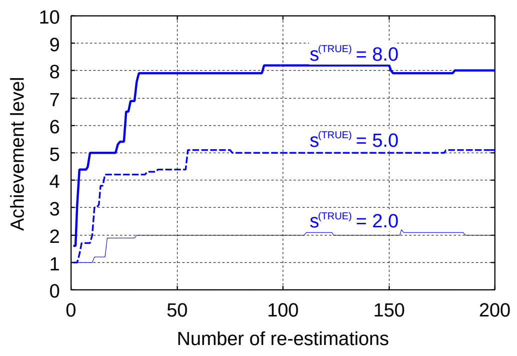

Figure 4: Variance of students' achievement level evaluated by AEGIS in our simulation

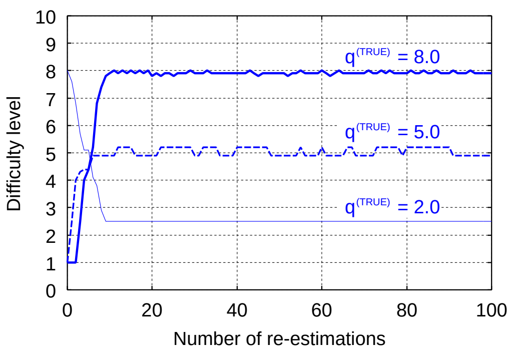

Figure 5: Variance of questions' difficulty level evaluated by AEGIS in our simulation

the *hidden region*, and calculates the level dynamically at regular intervals with the following formula:

$$q_{j,t} = \begin{cases} q_{j,t-1} + \dfrac{\displaystyle\sum_{i \in S} (s_{i,\tau} - q_{j,t-1})\xi_{i,j}}{\displaystyle\sum_{i \in S} \xi_{i,j}} & \text{if} \displaystyle\sum_{i \in S} \xi_{i,j} \neq 0 \\ q_{j,t-1} & \text{otherwise} \end{cases}$$

where S is a set of students who answered the question j between time $(t-1)$ and t, $s_{i,\tau}$ is a student's achievement level at time τ $(t-1 \leq \tau < t)$. The weight value, $\xi_{i,j}$, stands for 1 if students whose achievement level is more than the difficulty level $q_{j,t-1}$ of question j answered it wrongly or students whose level is less than $q_{j,t-1}$ answered it correctly, or 0 otherwise. This equation is a recurrence formula, and $q_{j,0}$, which is the initial difficulty level of question j, is given with the attribute LEVEL of DEL tag by teachers.

The above formula calculates a new difficulty level of the question for each question-type. Since the teachers assign the upper and lower limits of the difficulty level of a *hidden region* with the attribute LEVEL of DEL tag, AEGIS sets the upper (lower) limit as the initial difficulty level of the question to be generated in the error-correcting question (multiple-choice question). That of the fill-the-gap question is assigned the mean of the upper and lower limits. After calculating the new difficulty levels, AEGIS sets the maximum (minimum) value among the three-question types as the upper (lower) limit of the difficulty level of the *hidden region*.

4 Evaluation with Simulator

AEGIS estimates dynamically both the achievement level of each student and the difficulty level of each question with the equations defined in Section 3. In order to examine their validity, we investigated the following three things: 1. How does AEGIS estimate the inherent achievement level of a student? 2. How does AEGIS estimate the inherent difficulty level of a question? 3. Can AEGIS provide only questions suitable for a student?

We assumed that the probability that a student answers a question follows the function of both the inherent difficulty level of the question($q^{(TRUE)}$) and his/her inherent achievement level ($s^{(TRUE)}$), and that he/she can correctly answer the question with 50% probability if $q^{(TRUE)}$ is equal to $s^{(TRUE)}$.

1. Re-estimation of the achievement level of a student

   Based on the student's achievement level calculating formula defined in Section 3, we made AEGIS calculate the achievement level ($s^{(AEGIS)}$) of three students whose inherent achievement levels are high ($s^{(TRUE)} = 8$), middle ($s^{(TRUE)} = 5$) and low ($s^{(TRUE)} = 2$). We used 100 questions whose difficulty levels are distributed at the equal interval from 0 to 10.

   Figure 4 shows the variance of $s^{(AEGIS)}$. Since all of their levels, $s^{(AEGIS)}$, are set to 1 at first, the three curves in the graph start from 1. They were gradually separated each other because they answered questions

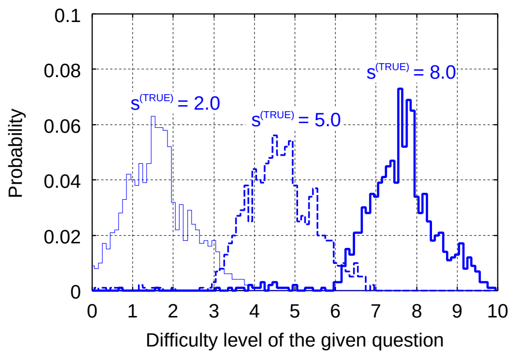

Figure 6: Distribution of the questions that AEGIS gave to students in our simulation

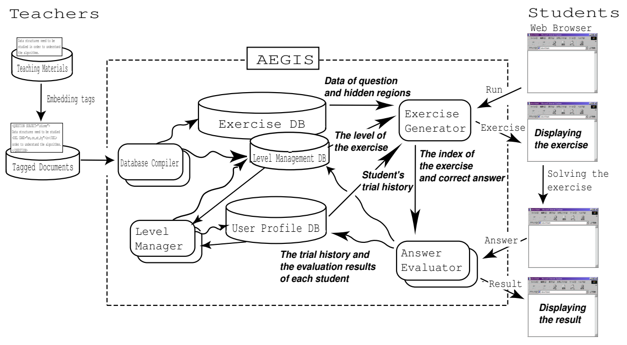

Figure 7: Overview of AEGIS

correctly or incorrectly based on their inherent achievement level. Each curve of $s^{(AEGIS)}$ in our simulation approximates closely the value of $s^{(TRUE)}$ that we presupposed. AEGIS can consequently distinguish between them.

2. Re-estimation of the difficulty level of a question

   Based on the question's difficulty level calculating formula defined in Section 3, we made AEGIS calculate the difficulty level ($q^{(AEGIS)}$) of three questions whose inherent difficulty levels are high ($q^{(TRUE)} = 8$), middle ($q^{(TRUE)} = 5$) and low ($q^{(TRUE)} = 2$). We at first set $q^{(AEGIS)}$ to a different value far from $q^{(TRUE)}$, that is, $q^{(AEGIS)}$ of the questions whose inherent levels were 5 or 8 was set to 1, and that of question whose inherent level was 2, was set to 8.

   In order to estimate the movement of the difficulty level of these questions, we used the two real distributions, called Class A and Class B, of achievement level of students who were given a lecture by one of the authors. Class A is superior to Class B. We assumed that the distributions did not change during our experiment.

   The curve of $q^{(AEGIS)}$ is shown in Fig. 5 for the data of Class A. Each curve of $q^{(AEGIS)}$ approximately converges to its inherent difficulty level $q^{(TRUE)}$. The students in Class A tried to solve a lot of questions of higher difficulty level. Although we would have expected that the current difficulty level of a question tried by many students would influence the change of the new difficulty level of a question, the result of our experiment on Class B becomes similar to the curve in Fig. 5. We can conclude that our method can well estimate the difficulty level of each question.

3. The difficulty level of questions generated by AEGIS

   We prepared three trial histories each of which belonged to high, middle, and low achievement level students. We applied 1,000 questions of various difficulty levels to these students so as to confirm that AEGIS generates good questions suitable for the students' achievement level. We assume that each 100 questions are uniformly distributed between difficulty levels 0 and 10.

   Figure 6 shows the distributions of the difficulty levels of questions which each student tried to solve. A student of high (resp. middle, low) achievement level tried a lot of questions of high (resp. middle, low) difficulty level. Let (x, y) be a pair of the mean value x and the standard deviation y of each distribution of the curve in Figure 6. (x, y) of each curve is (1.6, 0.78), (4.6, 0.87) and (7.6, 0.96), respectively. The result shows that AEGIS generates questions suitable for the students' achievement level.

5 Overview of AEGIS

The AEGIS system consists of three databases: *Exercise DB* (**EDB** for short), *User Profile DB* (**UPDB** for short) and *Level Management DB* (**LMDB** for short), and three main database managers: *Exercise Generator*, *Answer Evaluator* and *Level Manager*[Mine 2000]. The overview of AEGIS is shown in Fig. 7.

Teaching documents with the tags are compiled into the **EDB** and **LMDB**. All of the *question region*s are indexed sequentially and each *hidden region* is labeled with its own subindex of the index of each *question region*. The level of a *hidden region*, which is deeply related to the level of the question to be generated from the *hidden region*, is stored in the **LMDB** together with the index of the *hidden region*. The level of each *hidden region* in **LMDB** is reexamined regularly. **UPDB** keeps students' trial histories with their current achievement levels.

6 Conclusion and Future Work

We discussed our new Web-aided system AEGIS. The system is currently implemented in Perl scripts, PostgreSQL and CGI. AEGIS measures the student's achievement level every time he/she answers a question, and regularly re-estimates the difficulty level of questions in order to generate suitable exercises according to his/her achievement level. AEGIS is consequently utilized as not only a system generating exercises but also a tool classifying questions because the re-estimated level keeps close to their real difficulty level.

Experimental results with the simulator showed the effectiveness of the algorithm estimating both the achievement level of a student and the difficulty level of a question as were expected. We have a plan to evaluate this system by applying it to the real courses of Computer Literacy, which are taken by more than 2300 students at Kyushu University. We hope it will work fine as an educational tool for every student and help him/her to understand his/her subjects. Also, we will implement a tagging tool and an algorithm to generate another kind of exercise that allows more than one correct answers.

References

[Browning 1997] P. Browning, J. Williams, D. Brickley, and H. Missou, "Question Delivery over the Web using TML," http://www.ilrt.bris.ac.uk/netquest/liveserver/qbanks/demos/paul/lboro/ohp1.html, CAA, 1997

[Browning 1998] P. Browning, "TUTORIAL MARKUP LANGUAGE - A CBA SYSTEM," http://www.soton.ac.uk/ ukgec/workshop/5-cba/minutes.htm#TUTORIAL, 1998.

[Carbone 1997] A. Carbone and P. Schendzielorz, "A Web-Based Quiz Generator for Use in Tutorials and Assessment," Global J. of Engng. Educ., vol.I, no.3, 1997, http://www.eng.monash.edu.au/usicee/gjee/vol1no3/paper20.htm.

[ClassBuilder] ClassBuilder GradeBook And Exam Creation Software, http://www.classbuilder.com.

[Fujimoto 1999] R. Fujimoto, A. Suganuma, and T. Mine, "Development of a classroom management system on the web," Proc. Int. Conf. on Computers in Education, ICCE'99, vol. 2, pp.756–759, 1999.

[Hirokawa 1996] S. Hirokawa, T. Miyahara, T. Mine, T. Shoudai, and M. Mori, "Teaching 2300 students with www – practice and experience at kyushu university," Proc. ERI'96, pp.59–63, 1996.

[Mine 1998] T. Mine, D. Nagano, K. Baba, T. Shoudai, and S. Hirokawa, "On-web visualizing a mechanism of a single chip computer for computer literacy courses," Proc. Int. Conf. on Computers in Education, ICCE'98, vol. 2, pp.496–499, 1998.

[Mine 2000] T. Mine, A. Suganuma, and T. Shoudai, "The Design and Implementation of Automatic Exercise Generator with Tagged Documents based on the Intelligence of Students: AEGIS," Proc. Int. Conf. on Computers in Education, pp.651-658, 2000.

[Sato 1997] H. Sato, T. Mine, T. Shoudai, H. Arimura, and S. Hirokawa, "On web visualizing how programs run for teaching 2300 students," Proc. Int. Conf. on Computers in Education, ICCE'97, pp.952–954, 1997.

[Shoudai 2000] T. Shoudai, A. Suganuma, and T. Mine, "AEGIS: Automatic Exercise Generator with Tagged Documents based on the Intelligence of the Students," Proc. Knowledge-Based Software Engineering, pp.311–314, 2000.

[Suganuma 2000] A. Suganuma, R. Fujimoto, and Y. Tsutsumi, "An WWW-based Supporting System Realizing Cooperative Environment for Classroom Teaching," Proc. World Conf. on the WWW and Internet, pp.830-831, 2000.