Lamboudis, D. & Economides, A.A.: **Adaptive exploration of user knowledge in computer based testing.** *Proceedings 6th WSEAS International Conference on TELEINFO - Telecommunication Systems and Computer Science, CD-ROM, also in WSEAS Transactions on Communications*, Vol. 3, Issue 1, 2004.

Adaptive Exploration of User Knowledge in Computer Based Testing

DIMITRIOS LAMBOUDIS, ANASTASIOS ECONOMIDES
University of Macedonia
156 Egnatia Str.
GREECE
dimlamb@uom.gr, economid@uom.gr http://www.uom.gr

Abstract: - In the context of Computer Based Testing (CBT) and Computer Adaptive Testing (CAT) systems, two shortcomings can be identified. First, they seldom make use of information about how confident a student is in the answer given, and second, it is quite possible that students can get good marks by a combination of partial knowledge and guesswork [4]. More generally, in Computer Based Learning Environments, the representation and maintenance of user's knowledge can be considered as one of the critical factors that affect the system's effectiveness. In such systems the evaluation of user knowledge derives from tests and tasks that the system proposes to the user to accomplish. Thus, they inherit the limitations of testing, mentioned above. This paper describes an approach that refines assessment results through user knowledge exploration, incorporating probabilities. We argue that the proposed approach may lead to a better mapping of the assessment results to user knowledge.

Key Words: - Computer Based Testing, Computer Adaptive Testing, Intelligent Learning Environments, Pedagogical Agents.

1 Introduction

In recent years, with the expanding use of information technology in education, many tests have begun to be administered on computer. The Test-Delivery methods can be summarized in: *Computer Fixed Tests* which is a fixed-length fixed-form test, analogous to traditional paper-and-pencil testing; *Automated Test Assembly for Online Delivery*, which produces multiple test forms that are equivalent in some sense; and *Computerized Adaptive Tests,* which dynamically produce question sequences adapted to the individual learner [9].

For the purposes of this paper we will focus on adaptive tests. All these tests are administered following some testing algorithm, which is a set of rules specifying the questions to be answered by the examinee and their order of presentation. The basic underlying theory of most adaptive testing systems is Item Response Theory (IRT), where briefly, the questions' sequence is based on the probability of the individual examinee to answer the next question [12]. There are also approaches using Bayesian networks and numerous other research efforts trying to enhance the effectiveness of the computerized testing procedure. For example the incorporation of confidence testing, where the examinee is asked how confident she/he feels in answering a certain question before looking at the alternative answers [3], [13].

Although testing could be seen as a stand-alone procedure when used in traditional learning environments, e.g. universities etc., its importance is enhanced in the context of Intelligent Learning Environments (ILEs). These systems must consider a set of key decisions in their effort to support joint activity, including: *when* to engage learners with a service, *how* to best contribute to solving a problem, *when* to pass control back to users, *when* to query users for additional information, etc.

In order to reach such *situated* decisions, the system makes "guesses" about learners' needs, usually depending on the evidence obtained through the "keyhole" of the user interface, collaborative statistical data about the learner, explicitly asked information most commonly in the form of queries to the user in the beginning of a session, assessment evaluation, etc [5], [14].

The "intelligence" of a learning environment can be defined by its ability to make these decisions dynamically, at run- or user-time, based on an analysis of the learning context.

One of the main "ingredients" of the learning context is the learner and, from the system's point of view, the corresponding user model that the system maintains.

The student model stores information that is specific to each individual learner. At a minimum, such a model tracks how well a student is performing on the material being taught. Since the purpose of the student model is to provide data for the pedagogical module of the system, all of the information gathered should be able to be used in a meaningful way [10], [11].

In this paper we will focus on the system's capability to assess the current state of student's knowledge and the implied capability to do something "instructionally useful" based on the assessment. The learner's level of knowledge acquisition is evaluated by tests and/or tasks the user has to accomplish. That is, the responses of the user are mapped to its actual knowledge representation

One of the biggest challenges, in both stand-alone testing systems and intelligent learning environments, is to account for "noisy" data; the fact that students do not always respond consistently, particularly when their knowledge is fragile. Although different styles of scoring and mapping can be found, there is a common assumption made: a correct answer maps to knowledge, while a wrong answer maps to ignorance, faults, etc.

It can be argued, however, that this approach has two main shortcomings:

- In case of a correct answer, there is always a possibility that the user has answered by chance or at least he/she is uncertain about the answer chosen; it must be mentioned that the majority of the tests or the tasks in hand are, or could be seen, as multiple-choice questions; thus, with a question with five alternative choices, the possibility that a correct answer is the result of a guess is 20%; a possibility that cannot be ignored;
- In case of a wrong answer there is always a possibility that the user was mislead by factors irrelevant with his/her knowledge; for example, poorly designed questions, poor graphics in case the answer depended on them, etc.

Both cases lead to misconceptions about the actual user knowledge, which are difficult to be traced and revealed in the learning procedure to follow.

2 The Proposed Algorithm

The proposed algorithm attempts to overcome some of the limitations that were mentioned in the previous section. The algorithm is engaged during a multiple-choice test, or in a task with discrete steps or sub tasks.

Instead of proceeding to the next question or task when the user provides an answer, the algorithm engages an exploration module allowing the user to have a second chance or prove the validity of his/her answer. This second chance is not provided unconditionally, since this would be equivalent to just adding more questions or tasks in the original design of the test, leading to a prolonged test that might not be ideal in all cases. Instead, when the user responds to a question, the algorithm decides to explore the answer's correspondence to actual knowledge by some probability P_e, and not to explore it by some probability $P_m = 1-P_e$. Thus, in the "worst case", the system will behave "conventionally", i.e. like in the existing systems. However, there is a possibility, which is partially defined by the designer, at least as far as the initial value of P_e is concerned, that the system will give the learner a second chance. Yet, if this possibility is heavily depending on the initial value of P_e, it would be just another ad hoc intervention of the designer, lacking any adaptive characteristics.

Instead, the probability of exploring user knowledge (i.e. the definition of P_e), is determined by the system, through the algorithm which checks if this exploration has any affects on the learning procedure, that is, if it reveals user knowledge that was previously hidden. In case it does, it reinforces the value of P_e, and in case it doesn't it decreases it. In the long run, this means that independently of the initial values of P_e and P_m the system will favour the option that actually helps the learner and the system to have a better representation of what the user actually knows. The corresponding notation and assumptions are as follows:

- A testing procedure that can be represented by a set of *n* ordered Questions or Tasks, $Q=\{Q_i, i=1... n\}$;
- An initial value of P_e^0 (the corresponding $P_m^0=1-P_e^0$);
  Where $P_e^0 = P$ (explore user knowledge/ given an answer);
- *Map* is a function that maps the answer of the student to his/her knowledge representation;
- *Explore* is procedure that is engaged to clarify user Knowledge, and

- *Update* is a function that updates the values of P_e^i and P_m^i;

The pseudo code of the algorithm is described below and its flow chart in Figure 1.

Pose Q_i to the Student

Given an Answer from the Student

By (P_m^{i-1}) Proceed to Map
of this Answer to actual Knowledge or
By (P_e^{i-1}) Explore Students Knowledge
If (New Answer = Answer) then Map
Else Proceed to New Map
Update (P_e^i, P_m^i)

Proceed to Q_{i+1}

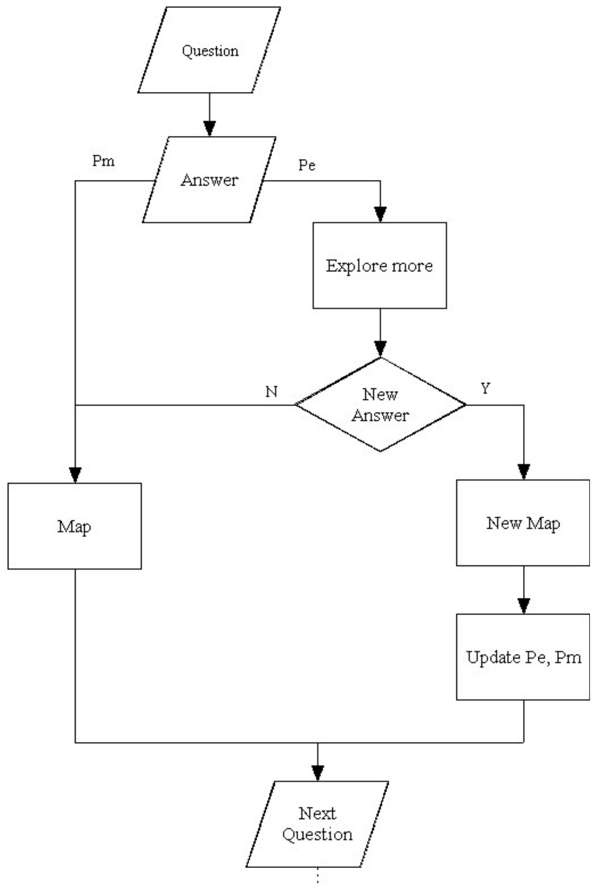

Figure 1. The algorithm Flow-Chart

Up to this point the algorithm actually describes the intervention strategy that the system follows in order to clarify possible misconceptions about the user knowledge. The "*Explore*", "*Update Probabilities*" and "*Map*", modules need to be further investigated. Although the functionality of these modules is still under research, we will present some ideas that were used in our preliminary implementation.

The "*Explore*" module is triggered by "chance" based on the probability computed in the "*Update Probability*" module presented next. In case the learner has responded correctly in the original question, it could ask for further details in the particular subject to check the validity of the original answer. In case the learner has responded with a wrong answer, it could pose the question in a different style. For example negatively posed questions quite often lead to misconceptions. The "Explore" module could rephrase the question in a positive manner, etc.

The depth of exploration, i.e. how many additional questions the module will invoke, is up to the designer and depends on the implementation. For example, in common multiple-choice tests one additional question could be adequate, while in a more complicated learning procedure more extensive exploration might be needed.

The "*Update Probabilities*" module controls the values of probabilities that trigger the exploring module. The updating strategy is based on the assumption that if the learner gives similar answers to both original and exploring questions these answers are consistent with his/her knowledge. Thus in the questions to follow the algorithm decreases the probability of exploration.

In the opposite case, where the original and exploration answers differ, it can be assumed that there is some kind of misconception or that the learner is not so confident about his/her answers. Thus the algorithm increases the probability of exploration in the questions to follow. In both cases we argue that the system will favor the option that is meaningful for the user and the system itself, literally adapting its behavior to the individual learner.

The exact values of the increment or the decrement as well as the initial values of probabilities depend on the designer's scopes. For example if we wish an initially neutral system we set $P_m = P_e =0,5$. Limitations should also be incorporated to specify the upper and lower limits for P_e^i and P_m^i, depending on the particular implementation. These limits will prevent, if necessary, the values of probabilities to reach unit or zero and lock the algorithm to continuously intervene and explore, or proceed with out exploring.

The "*Map*" module could be seen, at least as far as multiple question tests are considered, as a scoring module. Its task is to interpret the answers given by the student in both original and exploration questions in to a fair score. In case there is no extra information from exploration, i.e.

either the algorithm did not invoke at all with additional questions, or the learner was absolutely consistent in his/her answers, scoring can proceed as usual adding up the predefined partial scores for each question. In case there is extra information from exploration, a more detailed procedure need to be followed and it is up to the designer and the particular implementation to decide the scoring strategy. For example, if there is a direct conflict between an original and an exploring answer, the answer could be discarded or the learner could be granted with some scoring points depending on his/her overall achievement.

A simple example of the algorithm intervention will be presented to demonstrate its use.

3 Example

Let us assume that in the context of a preliminary computer science course the students are to be tested with a multiple choice test, part of which includes the following question:

1. *What of the following is wrong in the context of MS Windows Operational System:*

   a. A file name can have Greek characters
   b. Two files can have same names but different extensions
   c. Two files can have the same name and extension if they are stored in different directories.
   d. Two files in any case cannot have the same name and extension.

Suppose that the student selects one of the wrong answers in this question. Instead of proceeding to the next question the system invokes the algorithm. A uniform random number generator calculates a number *P,* between 0 and 1, which actually corresponds to a probability value. The result is checked against the current values of P_e and P_m. If $P < P_e$ the system will proceed to the next question; else the algorithm will engage the *explore* module. We remind that the underlying idea is that the student may know the correct answer but, for example, was mislead by the question's phrasing. For the purposes of our example we suppose that $P > P_e$ and exploration is triggered. An additional question is then presented to the student, equivalent to the original one. For example:

1.1 What is true for the filenames in MS Windows?

   a. A file name can have Greek characters.
   b. Two files can have same names but different extensions
   c. Two files can have the same name and extension if they are stored in different directories.
   d. All of the above.

If the student selects other than (d), that is, he/she continues to be wrong, we have good reasons to assume that he/she is not familiar with file naming in Windows. From the algorithm's point of view that means that there was no misconception and the student's answers are consistent with his/her knowledge. Thus additional questions *may* not provide any useful information in the following questions. If, in the opposite, the student selects answer (d), the correct answer, there is a clue of misconception in the original question. This misconception has derived, either from the question itself or from luck of confidence from the student's side. In any case it can be assumed that additional questions *may* be useful for the particular student in the questions to follow.

The results of exploring are then passed in to the *Update Probabilities* module. In the first case of similar answers, and based on the analysis that we have made, the algorithm will decrease the value of P_e. Thus, at least for the next question the probability of exploration will be reduced. In the second case of different answers, the algorithm will increase the value of P_e thus increasing the probability of exploration in the next question.

The *Map* module will evaluate the feedback provided by the user. A hypothetical scoring strategy is shown in table 1.

Answer to Q 1 / Answer to Q 1.1	**Wrong**	**Correct**
Wrong	0	0.5
Correct	0.5	1

Table 1. Scoring Strategy

In practice, scoring proved to be quite tricky in order to maintain its consistency among students. Generally defining the properties of the measurement scale, labelling the units, and interpreting the values derived are complex issues and require further research. [2]

Completing the question in hand the system will proceed to the next one, but this time with the new values for probabilities P_e and $P_{m.}$ The algorithm will be engaged in exactly the same manner and depending on the comparison of the outcome of the random generator and the new probability values will proceed to exploration. We argue that this iterative computation of probabilities and the corresponding biasing of the system's behaviour enhance the system's adaptivity.

Although we have focused in multiple-choice tests, it must be noted that in its general form the algorithm can be integrated in systems that use the assessment procedure to trigger intervention from the system's side. A very suitable example could be the case of a learning environment that is inhabited by an animated pedagogical agent [6]. In these environments the agent is physically present and one of its tasks is to monitor the learning procedure and act correspondingly. Misconception detection is one of the most tedious tasks and actually triggers most of the times the agent's intervention. Misconception is defined either as a deviation or a completely wrong answer-act from the user side, compared with the predefined "expert's choice". On the other hand a correct choice-act is apprehended as actual knowledge. We have argued that this approach has some shortcomings and that the proposed algorithm could be used to overcome them. Thus we argue that the algorithm could be integrated in such systems to enhance the "intelligence" of the agent's intervention strategy.

Moreover, this intervention will not only be useful to resolve misconceptions, but it will be done in a way that "hides" the behaviour pattern of agent, thus enhancing its believability [1], [7] [8]. This is achieved as result of the use of probabilities, which always leave a window of "chance" in the agent's behavior. It must be noted that the algorithm maintains the predefined pedagogical mainstream of decision-making, providing some low-cost additional information.

4 Conclusions

This paper proposes an algorithm that aims to enhance the systems ability to keep track of user's knowledge more reliably and more adaptively. Moreover, in case that it is used as part of the intervention strategy it could preserve the systems believability.

It can be also argued that although the proposed approach cannot formulate an autonomous procedure, it can be plugged – in to most testing delivery methods.

We have conducted some early experiments with students of our department to evaluate the algorithm. In particular we had our students run a simple multiple-choice test with and without the integration of the algorithm. This informal evaluation provided very positive results. Scoring was averagely 20% different, revealing lucky guesses but also not very clear questions.

Further work needs is currently under progress in the exploring and mapping modules in order to integrate a complete suite for assessment.

References

[1] Bates, J. (1994). The Role of Emotion in Believable Agents. *Communications of the ACM, 37* (7), 122-125.

[2] Crocker, Linda M., *Introduction to Classical and Modern Test Theory*. Wadsworth Group, 1986

[3] Davies, P. (2002), There's No Confidence in Multiple-Choice Testing…, Computer-Assisted Assessment Conference 2002 (CAA-2002), Loughborough, England.

[4] Gardner-Medwin, A.R. (1995) Confidence assessment in the teaching of basic science, ALT-J, vol. 3 no 1.

[5] Horvitz, E. (1999). Principles of Mixed-Initiative User Interfaces. *Proceedings of CHI '99, ACM SIGCHI Conference on Human Factors in Computing Systems, Pittsburgh, PA, May 1999.*

[6] Johnson W. L. (2000), Pedagogical Agents. *MIT Press*.

[7] Lamboudis D., Economides A., Managing Time Thresholds in Mixed Initiative Environments. *E-Learn 2002, World Conference on E-learning in Corporate Government, Healthcare and Higher Education, Montreal Oct. 15-19 2002*.

[8] Lester, J. C., Converse, S. A. Kahler, S. E. Barlow, S. T., Stone, B. A. and Bhogal, R. (1997a). The Persona Effect: Affective Impact of Animated Pedagogical Agents. *Proceedings of CHI '97 (Human Factors in Computing Systems,* pp. 359-366.

[9] Parsal Cynthia. [et al.]. *Practical Considerations in Computer Based Testing*. Springer-Verlag 2002 ISBN 0-387-98731-2.

[10] Sampson D., Karagiannidis, C., Kinshuk, (2002). Personalised Learning: Educational, Technological and Standardisation Perspective. *Interactive Educational Multimedia,* number 4 (April 2002), pp. 24-39.

[11] Shute V. and Psotka J., Intelligent Tutoring Systems: Past, Present and Future. *In Jonassen D. (ed.), Handbook of Research on Educational Communications and Technology. Scholastic Publications.*

[12] Wainer Howard [et al.]. *Computerized Adaptive Testing*. Lawrence Erlbaum Associates 2000. ISBN 0-8058-3511-3.

[13] Vomlel J., Bayesian Networks in Educational Testing, In Proceedings of First European Workshop on Probabilistic Graphical Models (PGM'02), November 6-8, 2002, Cuenca, Spain, pp. 176-185.

[14] Zuckerman I., and Albreciit D. W. (2001). Predictive Statistical Models for User Modelling. *User Modelling and User Adapted Interaction.*