Applying Multiple Student Modeling Techniques In Intelligent Tutoring Systems

Essam M. Kosba, Ahmed R. Dawoud
Arab Academy for Science & Technology / October University For Modern Sciences

Abstract

An important aspect of Intelligent Tutoring Systems (ITSs) is their ability to provide individualized instruction in a manner similar to what offered by a personal human instructor. A student model is described as the information that ITS keeps about an individual student. ITSs should actively support the student's learning process through tailoring the teaching process carried out to each individual student. The main purpose of a student model is to provide the planning component of an ITS with the information it needs to select a suitable instructional action. Probability Theory Intelligent Tutoring System (PTITS) is an intelligent system for teaching the fundamentals of the probability theory. The PTITS' approach to building the student model relies on gathering a great deal of information about the student through employment of both overlay and buggy models. An approach to inexact modeling of student ability based on certainty theory and fuzzy sets theory was adopted as a way to formulate the knowledge required in these models. More adaptability to the student status and more flexibility to diagnose student misconceptions are the main goals behind the conjunction of both models in PTITS. The developed architecture opens the door for more participation from teachers and instructors in developing their own courses using ITSs and hence for more conviction with ITSs' role in education.

1- Introduction

It is known that the development of any applied ITS is an extremely difficult and complex problem. This is because most of the developers start their ITSs from scratch, and therefore they have to build all of its complex parts, which take great effort and long time. In general, applied ITSs are developed on the basis of preliminary elaborated Expert Systems (ES) in the domain under study. These ES model the processes of problem solving in certain domain by an expert and thus represent Expert Models. Then student model is build upon ES. Finally, the pedagogical functions or Tutor Model is developed.

The main goal of this research is to build up an ITS that use both overlay and buggy student modeling approaches. Probability theory, as an important course in pre-engineering curricula, was adopted to present a domain for applying ideas of this research. The resulted system is called "Probability Theory Intelligent Tutoring System" or PTITS. The Knowledge base for probability theory and its problem solver are not available. So, a major technical consideration of our work is to lessen the complexity of the knowledge acquisition process and software engineering requirements involved in building of the domain knowledge base and problem solver without affecting our work in PTITS' student model. It is important to note that, when we simplify the knowledge engineering processes this will lead to more cooperation from the human teachers in building the ITSs and therefore increase the popularity of such systems.

2- Inference with Uncertainty

Building ITSs usually deals with a great deal of uncertainty especially when they try to model the student status. In this section we will shortly present the methods that can be used to acquire and represent the uncertain knowledge.

Probability and Bayesian Theorem: An important goal for many of the problem solving systems is to collect evidence as the system goes along and to modify its behavior on the basis of the evidence. To model this behavior, we need a statistical theory of evidence or Bayesian statistics theory. Bayesian theorem is a mechanism for combining new and existent evidence usually given as subjective probabilities [18]. It is used to revise existing prior probabilities based on new information. Bayesian approach can be explained as follow: if (E) is the evidence then each hypothesis (*H*) has associated with it a value representing the probability that H holds in the light of all the evidence E, derived by using Bayesian inference. This means that Bayes' theorem provides a way of computing the probability of a particular event given some set of observations we have already made. The fundamental notation of Bayesian statistics can be stated as that of conditional probability, $P(H \setminus E)$, which is the probability of hypothesis *H* given that we have observed evidence *E* [11]. It is necessary to take into account the prior probability of *H* and the extent to which *E* provides evidence of *H*. To do this, we must define a universe that contains an exhaustive, mutually exclusive set of H_i 's among which we are trying to discriminate. Then let

$P(H_i \setminus E)$ = The probability that hypothesis H_i is true given evidence *E*

$P(E \setminus H_i)$ = The probability that we will observe evidence *E* given that hypothesis *i* is true

$P(H_i)$ = A priori probability that hypothesis *i* is true in the absence of any specific evidence

k = The number of possible hypotheses.

Bayes' theorem then states that

$$P(H_i \setminus E) = \frac{P(E \setminus H_i).P(H_i)}{\sum_{n=1}^{k} P(E \setminus H_n).P(H_n)} \qquad (1)$$

Using Bayesian statistics, what we have inferred about a proposition is represented by a single value for its likelihood [18]. This leads to two criticisms; 1) single value does not tell us much about its precision, which may be very low when the value is derived from uncertain evidence and 2) the single value combines the evidence for and against a proposition without indicating how much there is of each. In fact, Bayes' theorem is intractable because the knowledge acquisition problem is very difficult; too many probabilities have to be provided. In addition, there is substantial empirical evidence that people are very poor probability estimators [7 19].

Reasoning using Certainty Factors: Standard statistical methods are based on the assumption that uncertainty is the probability that an event is true or false. In certainty theory uncertainty is represented as a *degree of belief*. In any non-probabilistic method of uncertainty one needs to go through two steps; 1) expressing the degree of belief and 2) manipulating degrees of belief during the use of knowledge-based systems. Certainty Factor (CF) can be defined as a figure that expresses belief in an event, fact, or hypothesis based on evidence or expert's assessment [18]. Several methods can be used to handle CFs in the knowledge-based systems. 1.0 or 100 can be used to represent absolute confidence and 0 for

certain falsehood. The medical diagnostic system MYCIN deals with uncertainties. In MYCIN, the numbers attached to CFs take values in the range (-1,1). If the value is positive one believes that the fact is true; if it is negative one believes that fact is not true, with complete certainty at each extreme -1 and +1 [9]. CFs can be used to combine different estimates of experts in several ways. Many researchers agreed that the most acceptable approach is used in MYCIN [1 13 14]. A certainty factor $(CF[h,e])$ is defined in terms of two parts; 1) $MB[h,e]$ - a measure (from 0 to 1) of belief in hypothesis *h* given the evidence *e*. *MB* measures the extent to which the evidence supports the hypothesis. *MB* is zero if the evidence fails to support the hypothesis. 2) $MD[h,e]$ - A measure (from 0 to 1) of disbelief in hypothesis *h* given the evidence *e*. *MD* measures the extent to which the evidence supports the negation of the hypothesis. *MD* is zero if the evidence supports the hypothesis.
Using these two measures, we can define the certainty factor as:

$$CF[h,e] = MB[h,e] - MD[h,e] \tag{2}$$

When several pieces of evidence are combined to determine the CF of one hypothesis, the measures of belief and disbelief of a hypothesis given two observations s_1 and s_2 are computed from:

$$MB[h, s_1 \wedge s_2] = 0 \qquad \text{If } MD[h, s_1 \wedge s_2] = 1$$
$$= MB[h, s_1] + MB[h, s_2] \cdot (1 - MB[h, s_1]) \qquad \text{Otherwise} \tag{3}$$
$$MD[h, s_1 \wedge s_2] = 0 \qquad \text{If } MB[h, s_1 \wedge s_2] = 1$$
$$= MD[h, s_1] + MD[h, s_2] \cdot (1 - MD[h, s_1]) \qquad \text{Otherwise} \tag{4}$$

This can be stated as: the measure of belief in *h* is zero if *h* is disbelieved with certainty. Otherwise, the measure of belief in *h* given two observations is the measure of belief given only one observation plus some increment for the second observation. This increment is computed by first taking the difference between complete certainty i.e., 1 and the belief given only the first observation. This difference is the most that can be added by the second observation. The difference is then scaled by the belief in *h* given only the second observation. Similarly, we can give an explanation for the formula of computing disbelief. Using *MB* and *MD*, *CF* can be computed.

The approach of CFs makes strong independece assumptions that make it relatively easy to use; at the same time these assumptions create dangers if the important dependencies are not captured correctly. This framework is useful and it appears that in an otherwise robust system the exact numbers that are used do not matter very much [11]. Another interesting thing about the CF approach is that it appears to mimic quite well the way people manipulate certainties [15].

Fuzzy Logic and Fuzzy Sets: Fuzzy logic is a superset of conventional or Boolean logic that has been extended to handle the concept of partial truth i.e., truth values between “completely true” and “completely false”. Dr. Lotfi Zadeh of UC/Berkeley introduced it in the 1960’s as a means to model the uncertainty of natural language. The motivation for fuzzy sets is provided by the need to represent propositions like “Ali is very tall”, “Ahmed is slightly ill”, etc. [11]. Traditional set theory defines set membership as a Boolean predicate e.g., one is either tall or not and there must be a specific height that defines the boundary. Fuzzy set theory allows us to represent set membership as a possibility distribution i.e.; one’s tallness increases with one’s height until the maximum boundary is reached. Some AI programs exploit the technique of inexact or approximate reasoning. This technique, which uses the mathematical theory of fuzzy sets, simulates the process of the normal human reasoning by allowing the computer to behave less precisely and logically than conventional computers do. The thinking behind this approach is that decision-making is not always a matter of black and white; it often involves gray areas. In contrast to certainty factors that include two values (degree of belief and disbelief), fuzzy sets use a spectrum of possible values. Fuzzy logic is difficult to apply especially when people provide evidence. The problems stem from linguistic vagueness to difficulties in supplying the definitions needed [18].

3- Major Components of ITS

ITSs are computer programs designed to transfer knowledge about a particular domain through the use of individualized learning sessions. ITSs are meant to simulate human tutors in a one-on-one environment rather than human teachers in a classroom environment. ITSs are among the most promising of emerging technological solutions for coping with professional instructor shortages. The ability to offer personalized instruction to individual students can enable ITS to successfully increase the effective supply of information technology instructors. While ITSs cannot totally replace human instructors, they can greatly reduce the amount of time instructors must spend with individual students. The focus on individualized learning is the primary difference between ITS and Computer-Based Training (CBT). Most of the researchers in the field of ITSs agreed upon the major intelligent components that usually constitute a typical ITS. These components as reported by [2 5 12 17 21] are:

1- *Domain Knowledge Base or Expert model*: keeping the subject or domain materials and the skills, or procedures that the system intends to teach to a student.
2- *Student Model*: representing a student's knowledge about the subject domain i.e., what the student does and does not know.
3- *Teaching Strategy Module or Instructional Model:* describing the tutoring or instructional strategies that will be used by the systems during the teaching processes.
4- *Intelligent Student Interface:* the actual presentation of text and graphics and also the acceptance of student's inputs are accomplished through this part of ITS. Defining and deciding the shape of computer-student interactions is the corner stone in designing ITSs.

Through the interaction of these models, ITSs are able to make judgments about what the student knows, what misconceptions he might suffer, and how well a student is progressing. However, several architectures are now developed to enhance the building process and the performance of ITSs and to help tailoring ITSs effectively.

4- Student Modeling Approaches

One of the major sources of intelligence in ITSs is the existence of the student modeling facility. Student modeling aims to model the individual student and exploiting the captured information to modify system interaction to best facilitates student's learning. Assuming that knowledge is belief, a student model is the system's beliefs about the student's beliefs. Thus it is important to capture student's understanding and misunderstanding of the course contents [6].
Simply, student model is a model of student's knowledge and capabilities [12 17]. Human teachers do an excellent job of judging the student's answers in the context of his assumed level of understanding and past learning behavior. Thus, the human teachers effectively adapt their instructions to the student's competence. For this reason, ideal ITSs should seek

information about what the student knows, his level of proficiency, his past learning behavior, and the presentation methods to which he responds best. Depending upon this information, ITS can select a suitable level and method of presentation and it can evaluate his responses in terms of the areas he knows well and those in which he is more likely to have misunderstandings [12].

Overlay Approach to Student Modeling: One approach to student modeling, introduced by Goldstien is the use of overlay model by which, perceived student knowledge is matched against the domain knowledge base, and areas of student understanding are flagged [12]. This means that student's knowledge is viewed in terms of tutor's domain knowledge. In overlay systems, the student's knowledge is treated as a subset of an expert's knowledge; the objective of the instruction is to establish the closest correspondence between the two. This means that student model is conceptualized by comparing student's behavior with that of an expert [6]. In the overlay model the student is represented by a relatively simple mechanism which supports inferencing about the student's cognitive state relative to ideal domain expert. For example, if domain knowledge is represented in the form of semantic network, then individual arcs in the network are flagged as the student exhibits an understanding of the relationship involved. On the other hand, if a production system is used, then individual production rule can be flagged whenever the student exhibits problem-solving behavior that employs them [17]. This gives a chance to easy comparison between what the student knows and what he should know. The overlay model works well where the goal is to strictly impart the knowledge of the expert to the student [6]. Using overlay model student errors will be interpreted as a lack of knowledge; the possibility that the student may have incorrect knowledge is not considered. As a result, an ITS using overlay model will only provide instructional materials that attempt to complete the missing knowledge [17]. It is important to note here that there is no plan to correct the student's incorrect ideas and this can be considered as the main disadvantage of the overlay modeling.

Buggy Approach to Student Modeling: Burton indicates that the unusual nature of children's arithmetic bugs suggests that no subset of the expert's knowledge could explain the incorrect procedures used by novices [3]. Therefore he suggested his buggy model which employs both correct and buggy rules that the student may follow. Understanding a student's error then becomes a task of finding a suitable combination of these correct and buggy rules that together would produce the same incorrect answer as was produced by the student. In a buggy model the student is not considered a mere subset of the expert, rather the student will possess knowledge potentially different in quantity and in quality from expert knowledge. A common technique for implementing buggy model is to represent explicit knowledge of misconceptions besides the representation of the expert knowledge. This means that buggy model can hold the student's knowledge and beliefs beyond the range of the expert model. A fixed collection of bugs is referred to as a "bug library". As the student progresses the buggy model can be updated in regard to the presence or absence of bugs known in the bug library. The inclusion of the bugs in the buggy model allows more understanding of the student than can be accomplished with a simple overlay model [6]. The buggy model is theoretically more complete but there are some disadvantages; 1) it is difficult to implement because system must represent knowledge structures other than the expert knowledge embedded in the ITS [17], 2) the task of inferring bugs from student's interactions poses problems [20], 3) model do not explain why the bugs have occurred [20], 4) Sleeman found that re-teaching was as effective as remediating specific bugs [16], and 5) other studies have found that even within a selected domain the bugs diagnosed vary greatly over schools, and classes studied [10].

Fuzzy Student-Modeling: Student modeling task is fraught with uncertainty that result from multiple sources of student mistakes, careless errors, and lucky guesses [8]. Fuzzy set theory is an approach for student modeling, aims at building imprecise diagnostic student models. In fact, researchers interested in student modeling seek intractable information because there are many sources of uncertainty in modeling student knowledge. Several approaches to making student modeling more tractable have been developed in recent years. One of these approaches is the model-tracing approach [4]. Model-tracing approach can be considered as an example of inexact modeling because it is based on the belief that useful student models do not need to be precise. Fuzzy set theory attempts to capture the notion that items can have varying degrees of membership within a set, as opposed to the standard view that an item either belongs or does not belong in a set. For example a student might have partial membership within the set of people who are expert in a particular skill. In ITSs, we think that the inexact cognitive student models are good. This is because, ITSs try to imitate the human teacher, and human teacher typically constructs an approximate model of any individual student. This sometimes leads the teacher to classify the students into few categories with respect to their learning levels. This approximate model still serves to guide instructor's pedagogical decisions.

5- PTITS: System Architecture Overview

The *Domain Knowledge Base* (DKB) of PTITS depends mainly on pre-stored tutoring materials instead of their automatic generation using some sophisticated knowledge representation method. There will be no problem solver for probability theory but pre-stored problems will be used to evaluate student status and diagnose his misconceptions and bugs. The final correct result for each problem is predetermined together with domain concepts required during solving. A list of domain concepts studied within the domain will be used to represent the ideal domain (instructor) model. A fuzzy relation among these domain concepts must be represented in DKB to show the necessity of all other concepts for teaching each separate concept. A pre-stored fuzzy set of bugs that possibly happen will also be stored to help facilitate the building of the student buggy model. These bugs will be investigated through the knowledge acquisition phase of the system design. Pre-stored correction materials and anti-bugs concepts will be used as remedial actions in the case of bug occurrence.

The *PTITS' student model* consists of two major models: the Student Knowledge Model (SKM) and the Student Buggy Model (SBM). Each of these models will be divided into sub-models with a specific function for each one. The knowledge required for student modeling would be extracted from student actions during interaction with the system. We relied upon two main sources of information about which aspects of students' understanding and performance should be modeled by the system: analysis of student tasks and human-teacher judgments. The variables derived from these sources will be used as an indicator about some characteristics of student capability. Most of the knowledge handled by the student-modeling component depends mainly on the ideas of certainty theory and fuzzy sets. The main justification to this selection is that no human or computer tutor can exactly build a precise model for a student. Other justifications include the simplicity of these techniques and its adequacy for tutoring applications.

Everything that a teacher (human or machine) can infer about a student's knowledge and misconceptions is conveyed through the student actions. No user modeling system can peer directly into student's mind. Language and actions are the sole media through which modeling information passes. In PTITS student actions serve as the sole window through which diagnostic information about the student is conveyed. For example, when a student selects to work in a certain lesson and the system starts presentation of screens that contain domain materials or example problems, PTITS will keep important relative information in the student model. This information indicates what concepts are presented to the student and how much time each concept is presented. Also, the time student spent in reading materials, related to a specific concept, would be calculated to be compared with the optimum time determined by the domain expert. Another example, when the student is in the quiz phase, where the system diagnoses student's knowledge, and he selects an answer for a given problem, his selection will be translated as concept understanding, misconception, bug avoidance, or bug occurrence.

Teaching strategy in PTITS can be considered as a mixture between the normal tutoring approach (frame-based) and the coaching approach. A student can either follow the sequence of screens presented by the system or selects his desired sequence. In the first case (tutoring approach), the system will have the full control over the student and the available selections. In the second case (coaching approach) the student can select his own scenario and the system will observe his actions and interfere in appropriate times e.g., to give some explanation or remedial actions. The PTITS' teaching scenario can be summarized in the following three cycles described below and shown in Figure (1):

CYCLE-1: The cycle of active knowledge presentation. In this cycle, it is assumed that PTITS' tutor will present domain materials and explanatory examples related to one of the domain's lessons. The student can transfer between different screens and can repeat or skip screens, as he needs. In this cycle there is no need for checking student knowledge. An active knowledge presentation to students includes task such as selecting teaching material, presenting it to student, evaluating student possible selections and actions, or maintaining student model.

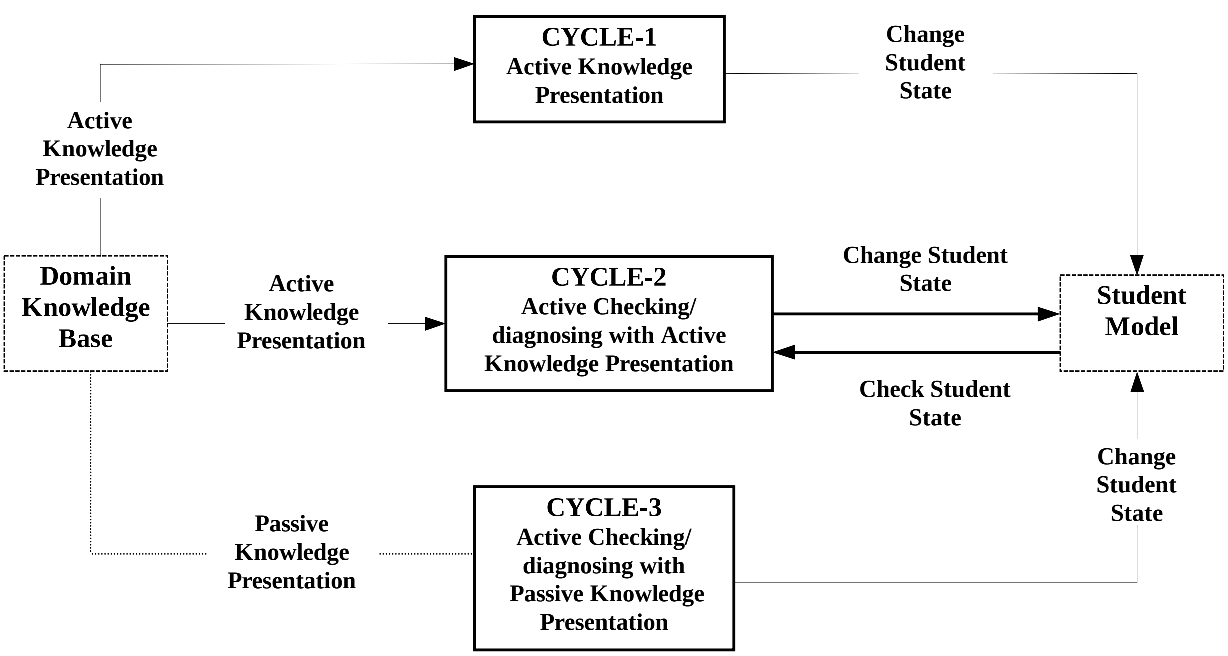

Figure (1): PTITS' Teaching Scenario

CYCLE-2: The cycle of active checking/diagnosing with active knowledge presentation. In this cycle it is assumed that the system will present the checking or diagnosing materials that can help PTITS to modify the student model in reaction to student answers. Here, knowledge delivery is active since system must provide the student with appropriate remedy. The type of comment and/or remedial action will depend on the diagnosing of the student current selection and also on the result of active diagnosing of student knowledge using the information stored in the student model. CYCLE-2 includes tasks like selecting diagnosing material, presenting diagnosing materials to student, evaluating student reactions, commenting student answers and his problem solving criteria, or maintaining student model.

CYCLE-3: The cycle of active checking/diagnosing with passive knowledge presentation. In this cycle PTITS will presents an exam to the student. It is assumed that the test material will serve as checking or diagnosing materials. This material can also help us to modify the student model according to student answers. Of course, in this testing cycle, the knowledge delivery is passive; there will be no commenting or remedy. The main objective of this cycle is to evaluate the student's performance at some predetermined places in the course. This evaluation will be translated to some sort of pointing or student grading. This cycle may include tasks like selecting diagnosing material (an exam), presenting test materials, evaluating student reactions and error identification, scoring or grading the student, or maintaining student model.

6- PTITS: Model Details

In last section we presented the outline that guided our work in the design of PTITS components. The PTITS system architecture is shown in the Figure (2). In this section, the components of PTITS are described in detail along with the criteria and procedures used.

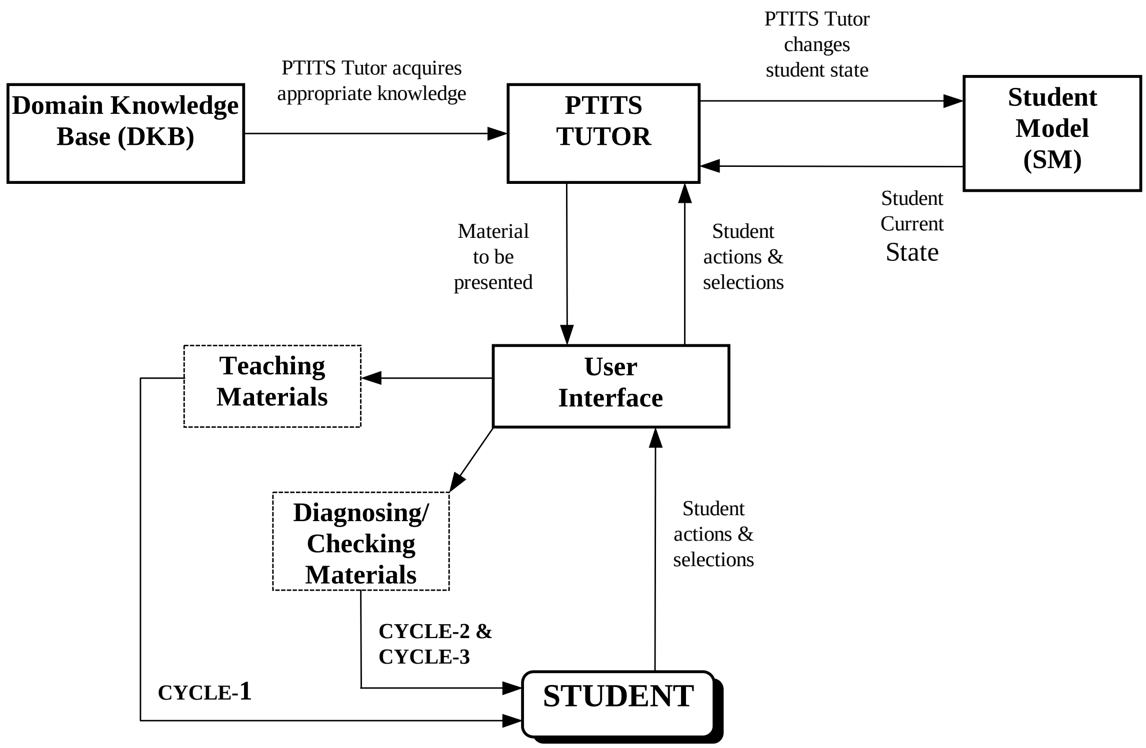

Figure (2): PTITS System

The PTITS Domain Knowledge Base (DKB): As stated before, DKB depends mainly on different types of pre-stored domain materials. Some of these materials are used to build the student's knowledge about the domain and others are used for student evaluation. *Pre-stored Instructional Materials* are represented by a sequence of screens that contain the body of the knowledge constitute the probability theory course. One or more screens may be used to define a single domain concept. In other cases, a single screen may contain materials belong to more than one concept. When PTITS presents a material or example screen and when student decide to end this presentation, the system will change the student modeling variables to indicate up to what level student go through the material presented for certain concept. The appropriate changes in the student model will not take place unless student spent a certain amount of time before he ends displaying the screen. Human expert (Instructor) predetermines this amount of time. When a student selects a certain lesson, the tutor will start directly to present the screens that contain the materials about the concepts related to the selected lesson.

By the end of these screens the PTITS tutor will start the presentation of the screens that contain examples related to materials just described. Students can go directly to example screens without going through the material screens. *Pre-stored Quiz Problems* are designed to help evaluating students. The student can choose to start quiz without reading material and/or examples, but PTITS will not allow a student to start, for example, lesson-2 unless he works on at least one quiz related to lesson-1. This is because the quiz phase is the place where PTITS performs student diagnosing. Quiz problems are designed to help PTITS checking the level of student understanding and also to test the student willingness to make certain bugs. *Pre-stored Exams* are used also to check student's status. A bank for different types of exams (7th, 12th, and Final) and also for different levels (easy, moderate, and hard) is prepared. The exams will be selected randomly by the system and according to pre-specified difficulty level. During the exam, a student can go from one question to another and can change his answers during the allowed time for the exam. The final answers delivered by the student should affect the student model. *Concept Table (CT)* contains the set of concepts that constitute the whole course. Fuzzy relations between these concepts are indicated using this table. For each concept, CT contains concept identification, concept description, and identifications of the most related three concepts along with their fuzzy relations to the current concept (high, moderate, or low). *Bug Table (BT)* contains the bugs that the student may fall in through the course. These bugs are acquired from the domain expert during the knowledge acquisition phase. BT contains bug identification, bug description, and concept identifications of the most related three concepts expert believes that, when not grasped, they cause the bug.

PTITS Student Model (SM) contains several sub-models. *Student Profile Model (SPM)* is designed to keep general information about the student. SPM contains general description of some cognitive and psychological characteristics and preferences of a student. SPM contains also some historical information that describes how the student go through the course e.g., number of sessions he made, the time elapsed in each session, his score in quizzes, etc.

Student Knowledge Model (SKM) is used to know which concepts are mastered by the student and which are not (overlay model). This helps the system to select the appropriate material that will cope with the student status. For each concept, SKM keeps information about the status of the student in relation to that concept. This information is stored in two parts:

1) Concepts SKM (CSKM): a record for each concept is created to indicate student's status. This record contains student identification, concept identification, Number of Concept Presentations (CPNO) - the accumulative number of presentations related to the concept, Concept Presentation Time (CPT) - time student spent in reading the presented materials, Concept Understanding Confidence (CU-CON), Number of Correct Answers (CCANO), Correct Answers Confidence (CCA-CON), Number of Misconceptions (MISCNO), and Misconception Confidence (MISC-CON). For the presented material or example related to a certain concept, the domain expert assigns a confidence level (from 0 to 1). This confidence level indicates the measure of belief (MB) that the student can understand the concept from this material. CU-CON is designed to hold the aggregate measure of belief that the student understands the concept after presentations he made. Assume that the system indicates the presentations shown in the Table (1) to a student when he studied a certain concept. Using this information system can make the following calculations:
CPNO = 4 times
CPT = 1.5 + 3 + 4 + 2 = 9.5 minutes
CU-CON (after presentation no. 1) = 0.4
According to equation (3)
CU-CON (after presentation no. 2) = 0.4+ 0.6 (1-0.4)= 0.76
CU-CON (after presentation no. 3) = 0.76+ 0.3(1-0.76)= 0.832
CU-CON (after presentation no. 4) = 0.832+ 0.8(1-0.832)= 0.9664
Now, we have a measure of belief equal to 0.9664 that the student understood the concept from the shown presentations.

Presentation no.	Presentation time (minutes)	Understanding MB
1	1.5	0.4
2	3	0.6
3	4	0.3
4	2	0.8

Table (1): CU-CON calculation example

When PTITS asks the student to solve problem related to a concept and the student selects the correct answer, the CCANO will be incremented by one. For each problem there is a measure of belief associated with the selection of the correct answer. This confidence is determined by the domain expert and indicates the measure of belief that the concept is understood by the student when correct answer is selected. CCA-CON will hold the cumulative confidence that the concept is understood by the student after the total number of correct answers he made and can be calculated using (3) as for CU-CON. When student selects a wrong answer, this will be treated as a misconception and MISCNO will be incremented by one. For each problem there is a measure of belief associated with the selection of any available wrong selection. This confidence level indicates the measure of belief that the concept is misunderstood by the student if this wrong answer is selected. MISC-CON will hold the cumulative measure of belief that the concept is misunderstood by the student after the total number of wrong answers he made.

2) Historical state of CSKM (HCSKM): The purpose of HCSKM is to keep the historical or detail behaviors of the student. A record in HCSKM is created each time the student selects an answer for a question regardless of the answer being correct or not. The record contains Student Identification, Concept Identification, Serial Number, Today's Date, and Concept Status (OK for correct answer or NOT OK for wrong answer).

Student Buggy Model (SBM) is designed to indicate bugs student made during his problem solving sessions and its frequency. For each bug in the BT, SBM should keep information about the status of the student in relation to that bug. This information is stored in two parts:

1) Occurrence SBM (OSBM): a record for each bug is created to hold the information that describes the status of the student. A single record in OSBM contains student identification, bug identification, Number of Bug Occurrences (BONO) - the number of times a student selects answers that can be considered as bug occurrence, Bug confidence level (BO-CON), Number of Bug Avoidance (BANO), and Bug Avoidance Confidence Level (BA-CON). In specific problems, some of the available answers will be designed to indicate bug occurrence. This means that the selection of a certain wrong answer may indicate a certain bug occurrence with specified confidence level. BO-CON holds the cumulative confidence level or measure of belief that the student suffers from this bug after the total number of buggy answers he made. As an example, assume that a student made the buggy selections shown in Table (2), hence system will perform the following calculations:
BONO = 3, and
BO-CON after first buggy selection = 0.6
According to equation (3)
BO-CON after second buggy selection = 0.6 + 0.8(1-0.6) = 0.92
BO-CON after Third buggy selection = 0.92 + 0.3(1-0.92) = 0.944
This means that we have a measure of belief equal to 0.944 that the student is suffering from a specific bug.

No. of buggy selections	Individual bug confidence or bug MB
1	0.6
2	0.8
3	0.3

Table (2): BO-CON calculation example

BANO indicates how many times a student selects the correct answer when he is required by the system to solve a problem with buggy selections. The selection of a correct answer in this case can be interpreted as bug avoidance with a certain confidence level. BA-CON holds the cumulative measure of belief that the student knows the bug and can avoid it. BA-CON can be calculated using (3) as in the case of BO-CON.

2) Historical SBM (HSBM): The purpose of HSBM part is to keep the historical or details behaviors of the student when he is checked in a certain bug. A record in HSBM is created each time student select a buggy choice or avoid it. The record contains Student Identification, Bug Identification (B-ID), Serial Number, Today's Date, and Bug status (OK for bug avoidance or NOT OK for buggy selection).

Remedial action (RA) is the piece of knowledge that PTITS decides to present as a means for the correction of a certain bug, or misconception. Amount of details and shape of this piece of knowledge depend mainly on the information retrieved from the student model. PTITS tutor selects the appropriate RA as one of seven types. *TYPE-1-RA;* requires the presentation of the complete concept materials along with the materials of related concepts according to their type of relations and their understanding states. *TYPE-2-RA;* requires the presentation of the complete concept materials along with the materials of only highly related concepts according to their understanding states. *TYPE-3-RA;* presents a short correction message just to remind the student with a certain concept. *TYPE-4-RA*, presents an example that is relatively similar to the problem in which a student made the mistake. *TYPE-5-RA;* takes place when student selects a wrong answer and his model demonstrates a high learning level of the related concept. *TYPE-6-RA;* is used when a student demonstrates some serious buggy behaviors. In this case, the complete domain material of the related concepts is required according to its understanding states. *TYPE-7-RA;* is used when the student demonstrates some moderate buggy behaviors. PTITS, in this case, presents a short anti-bug message to help student avoid the bug in the next problems.

Judging Student Status: It is important to explore the way used by PTITS to judge student status. These judgments help the system to select and present the appropriate RA.

1) Judgments from CSKM: In CSKM there are three confidence levels: CU-CON, CA-CON, and MISC-CON. CU-CON and CA-CON can be used together to determine the student's understanding measure of belief (UMB). Since CU-CON depends on the presentation evidence, and CA-CON depends on the correct answer evidences, then we can not consider both as equal factors in UMB calculation. This is because the evidence that a student understands the concept from correct answers is more certain than the evidence from just presenting and reading materials. Hence, PTITS compute UMB as a weighted-average of CU-CON and CA-CON. The weight given to the CU-CON is 1, while the weight given to CA-CON is 4. This means that we belief in the quiz measure four times more than our belief in the presentation measure. In traditional classrooms, human-teachers assign all the weight to the quizzes and exams in their judgments about the student. The selection of 4 here does not mean that this is an optimum selection, this is just a selection for our implementation. In our case then, we can calculate the UMB as:

$$\text{UMB} = (\text{CU-CON} + 4\ \text{CA-CON})/5$$

MISC-CON can be used directly to represent the understanding measure of disbelief UMD:

$$\text{UMD} = \text{MISC-CON}$$

Then according to (2), the understanding certainty factor (UCF) can be calculated as

$$\text{UCF} = \text{UMB} - \text{UMD}$$

There are five knowledge states for UCF: *completely unlearned*, *unlearned*, *semi-learned*, *learned*, and *completely learned*. These states represent the concept understanding fuzzy sets. Student may be assigned to one of these sets according to the value of UCF:

$0 \le \text{UCF} < 0.2$	Concept is completely unlearned
$0.2 \le \text{UCF} < 0.4$	Concept is unlearned
$0.4 \le \text{UCF} < 0.6$	Concept is semi-learned
$0.6 \le \text{UCF} < 0.8$	Concept is learned
$0.8 \le \text{UCF} \le 1$	Concept is completely learned

UCF is also used to determine the type of RA in the different cases as shown in the Table (3).

Knowledge state	Remedial Action

Completely Unlearned	TYPE-1-RA
Unlearned	TYPE-2-RA
Semi-Learned	TYPE-3-RA
Learned	TYPE-4-RA
Completely Learned	TYPE-5-RA

Table (3): Knowledge states and corresponding remedial Action

2) Judgments from OSBM: There are two confidence levels in OSBM: BO-CON and BA-CON. BO-CON indicates the measure of belief that the student is suffering from the bug. BA-CON, in other hand, indicates the measure of belief that the student can avoid the bug. As a consequence, PTITS calculates the Bug Certainty Factor (BCF) as follow:

Since Bug Measure of Belief (BMB) = BO-CON, and
Bug Measure of disbelief (BMD) = BA-CON
then Bug Certainty Factor (BCF) = (BO-CON) - (BA-CON)

There are three bug states: *assured*, *semi-assured*, and *no-bug.* These states represent the bug existence fuzzy sets. The student may be assigned to one of these sets according to BCF:

$BCF = 0$ There is no-bug
$0 < BCF < 0.5$ Bug is semi-assured
$0.5 < BCF \leq 1$ Bug is assured.

In the case of no-bug, there is no RA required. TYPE-7-RA is used in the case of semi-assured bug while TYPE-6-RA is used when the bug is assured.
3) Judgments form HCSKM and HSBM: One of the objectives behind the use of HCSKM and HSBM parts is to evaluate the student's behavior over time or, in other words, to measure the stability of knowledge in his mind. For example, if the student's answers related to certain concept during a long interval are "OK", then this means that the student is able to retain the knowledge for a long period. If the student answers are "OK" for a relatively short period (say, one week) and changed to "NOT OK" after relatively long period (say, 3 or 4 weeks), then this means that the student is unable to retain the knowledge for a long period. Weighted-average is a suggested analytical method to analyze the historical data generated in HCSKM and HSBM.

7- PTITS' Model Advantages and Limitations

PTITS' Model Advantages: It is clear that most of declarative models that constitute the PTITS domain knowledge base and student model are easy to build up by different experts. They set both expert and ITS designer free from a traditional very difficult design of procedures, plans, or rules of ITS functions. Using these declarative models, in addition to some procedures (e.g., to control pre-stored material presentation), and criteria (e.g., to evaluate and maintain the student model) the system can avoid problems related to the generation of teaching materials and evaluation tasks. It can also avoid problems related to selection of consequent teaching material for remedy purposes. One of the common goals today among researchers of ITSs is to simplify the enormously sophisticated procedures of new applied ITSs design. This basis helps to elaborate ideas of effective and inexpensive ITSs to become popular. Our suggested architecture facilitates more participation from the human teachers in designing the ITSs, so their beliefs in ITSs will be increased and more cooperation from their sides will be gained. PTITS in this way has a general modular architecture depending on the idea of inexact or uncertain student modeling. The techniques we used to update student model and draw inferences from its variables are far simpler computationally than those used in probabilistic approaches. One final advantage in PTITS is the ability to use the information generated in the different parts of the model for the research purposes. It can be used to determine which concepts are difficult for the majority of the student and which bugs are common between them. This of course should open the way for: 1) Enhancing and/or changing the ways used to explore these difficult concepts, and 2) Identifying the reasons behind the common bugs; does it come from defects in the course itself or from previous courses in past education stages? If we have the reasons, then we can find the solutions.

PTITS' Model Limitations: Actually, there are some limitations in PTITS model: 1) The knowledge base depends mainly on pre-stored materials as used in traditional CAI systems. These pre-stored materials affect negatively the system's intelligence. In fact, we decided to use such type of pre-stored materials to facilitate more participation from teachers' side in the development of ITSs and also to assist presenting our ideas related to student modeling. 2) There is no Expert Model designed for probability theory domain. So, the implemented version of PTITS can not automatically solve probability theory problems; it depends on the pre-stored answers for these problems.

8- PTITS: Rapid Prototype

A rapid prototype for the PTITS is developed to prove the feasibility of our ideas. The first lesson of the probability theory domain is prepared and represented in this prototype. The examples and different quizzes for this lesson are also prepared. A complete set of screens is designed to work as remedial actions for the cases of misconceptions and bug occurrence. PTITS' rapid prototype is developed using normal RDBMS (MS ACCESS). It contains two main parts: *1) The Instructor Panel:* from which an instructor can enter all the necessary knowledge required to build up the DKB, for examples, concepts, bugs, teaching materials, examples, quizzes and its answers, and remedial actions. The data required by GSPM can also be entered through this panel. The instructor panel holds the reporting capabilities that may be added to the system in other implementations. *2) The PTITS Tutor or Student Panel:* In which the actual teaching process takes place. The student can view, after passing the authority check, which parts he passed and which parts he did not. The PTITS Tutor gives the student a chance to select any part (lesson, topic, examples, or quiz) even it was selected before. Most of the mentioned student diagnosing rules and actions are implemented in this part of the system. Student modeling variables are used effectively to determine the suitable remedial actions that fit with the student current state.

9- Conclusion

Student modeling is constrained by both the popular approaches used in learning theory and artificial intelligence technology. Student modeling is a key feature to give both instruction and adequate help in teaching and learning environments. Much of the original work in student modeling was driven by work with overlay and buggy models. To be effective, student models must provide descriptions of the learner's understanding and misunderstanding at the level of granularity that facilitates effective instruction.

Our aim has been to develop ITS that uses both overlay and buggy models to evaluate and diagnose the student status. We have presented a prototype of PTITS able to build an internal multi-model of each individual student, representing his knowledge and misconceptions. There is no Knowledge Base or Expert Model available for probability theory domain, so

PTITS' knowledge base is built by using pre-stored instructional materials, examples, and quizzes. An approach to inexact modeling of student ability based on certainty theory and fuzzy logic was adopted as a way to formulate the knowledge required in these models. Some technical concerns such as simplifying the knowledge engineering process required to develop and maintain the student model, and decreasing the programming complexity are highly considered. Our main contribution so far has been in using the knowledge from both overlay and buggy models to individualize instruction according to the student status. Other contribution is that we can use relatively simple procedures to dynamically update fuzzy or uncertain distributions, which represent student competence on discrete knowledge components. The adequacy of certainty factor approach used in this research await evaluation to determine up to what level it is appropriate for ITSs applications. In evaluation process it is important to consider dimensions like knowledge engineering efforts needed to build student-modeling component, the complexity level of implementing and maintaining the model, and the extent to which human teacher can help in building the model.

Bibliography

1. Buchanan, B., and Shortliffe, E. (eds.), (1984). Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. *MA: Addison-Wesley.*
2. Burns, H., Parlett, J., and Redfield, C., (1991). Intelligent tutoring Systems: Evolutions in Design. *Lawrence Erlbaum Associates, Publishers.*
3. Burton, R., (1982). Diagnosing Bugs in a Simple Procedural Skill. In Sleeman, D., and Brown, J. (eds.), Intelligent Tutoring Systems. *Cambridge, MA: Academic Press*, pp. 157-183.
4. Derry, S., and Hawkes, L., (1992). Toward Fuzzy Diagnostic Assessment of Metacognitive Knowledge and Growth. *Paper presented at the annual meeting of the American Educational Research Association, San Francisco, CA.*
5. El-Gamal, Y., and El-Maghraby, A., (1994). An intelligent Tutoring System for Maritime Technology. *Proceedings of 8th Congress of International Association of Institute of Navigation (IAIN), Pekin, China, September 6th,* pp. 23-29.
6. Holt, P., Dubs, S., Jones, M., and Greer, J., (1994). The State of Student Modeling. In Greer, J., and McCalla, G. (eds.), Student Modeling: The Key To individualized Knowledge-based Instruction. *Springer-Verlag Berlin Heidelberg,* pp. 3-35.
7. Kahneman, D., Slovic, P., and Tversky, A., (eds.), (1982). Judgment Under Uncertainty: Heuristics and Biases. *New York: Cambridge University Press.*
8. Katz, S., Lesgold, A., Eggan, G., and Gordin, M., (1994). Modeling the Student in SHERLOCK II. In Greer, J., and McCalla, G. (eds.), Student Modeling: The Key To individualized Knowledge-based Instruction. *Springer-Verlag Berlin Heidelberg*, pp. 99-125.
9. Klein, M., and Methlie, L., (1995). Knowledge-based Decision Support Systems. *John Wiley & Sons Ltd.*
10. Payne, S., and Squibb, H., (1991). Algebra Mal-rules and Cognitive Accounts of Error. *Cognitive Science*, 14:445-481.
11. Rich, E., and Knight, k., (1993). Artificial Intelligence. *McGraw-Hill, Inc.*
12. Rickel, J., (1992). Intelligent Computer-Aided Instruction: A survey organized around system components. In Stephen, A., and Hopple, G. (eds.), Applied Artificial Intelligence: A source book. *McGrow-Hill, Inc*., pp. 205-234.
13. Shortliffe, E., and Buchanan, B., (1975). A Model of Inexact Reasoning in Medicine. *Mathematical Biosciences*, 23:351-379.
14. Shortliffe, E., (1976). Computer-Based Medical Consultation: MYCIN. *New York: Elsevier.*
15. Shultz, T., Zelago, P., and Engleberg, D., (1989). Managing Uncertainty in Rule-Base reasoning**.** *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society.*
16. Sleeman, D., Kelly, A., Martinak, W., and Moore, J., (1989). Studies of Diagnosis and Remediation with High School Algebra Students. *Cognitive Science*, 13:467-506.

17. Stephen, A., and Hopple, G. (eds.), (1992). Applied Artificial Intelligence: A source book. *McGrow-Hill, Inc.*
18. Turban, E., (1993). Decision Support and Expert Systems: Management Support Systems. *Macmillan Publishing Company, a division of Macmillan, Inc.*
19. Tversky, A., and Kahneman, D., (1974). Judgment Under Uncertainty: Heuristics and Biases. *Science*, 185:1124-1131.
20. Verdejo, M., (1994). Building a Student Model for an Intelligent Tutoring System. In Greer, J., and McCalla, G. (eds.), Student Modeling: The Key To individualized Knowledge-based Instruction. *Springer-Verlag Berlin Heidelberg*, pp. 148-163.
21. Zhou, G., Wang, J., and Ng, P., (1996). Curriculum Knowledge Representation and manipulation in Knowledge-Based Tutoring Systems**.** *IEEE Transactions on Knowledge and data Engineering,* 8:679-689.

ESSAM KOSBA
Essam Kosba is currently the manager of Information and Documentation Center of Arab Academy for Science, Technology, and Maritime Transport (AASTMT). He received his B.S. degree in Computer Engineering from Alexandria University – Egypt. Before receiving his MBA degree in MIS, he worked five years in Jordan as computer instructor, eight years in Egypt and Kingdom of Saudi Arabia as a systems analyst. He worked also as a head for Communications & Computing systems in Integrated Simulators Complex of AASTMT. Now, he is preparing his proposal for Ph.D. degree.