Large-Scale Deployment of Three Intelligent Web-based Database Tutors

Antonija Mitrovic and the ICTG team
Computer Science Department, University of Canterbury
Private Bag 4800, Christchurch, New Zealand
http://www.cosc.canterbury.ac.nz/~tanja/ictg.html

Abstract. *We present our experiences with DatabasePlace, a Web portal aimed at university-level students enrolled in database courses. The portal was established by Addison-Wesley in January 2003. Besides presenting information about the textbooks, the portal also provides additional domain information, online quizzes and three Intelligent Tutoring Systems developed by the Intelligent Computer Tutoring Group (ICTG). We briefly present the three systems, and then discuss our experiences. We also compare the DatabasePlace students to our local students using the three ITSs.*

Keywords. Intelligent Tutoring Systems, evaluation, student modelling

1. Introduction

Intelligent Tutoring Systems (ITS) are knowledge-based systems that provide individualized instruction, by being able to adapt to the knowledge, learning abilities and needs of each individual student. ITSs offer many advantages over the traditional classroom scenario: they are always available, non-judgemental and provide tailored feedback [1,11]. The current state-of-the-art ITSs achieve improvements of one standard deviation compared to traditional classroom teaching, but are not yet as effective as one-on-one human tutoring [1,2,7,8].

In this paper, we discuss our experiences in providing three Web-enabled ITSs that teach various database skills to university students. We have developed the three systems within ICTG, and have been using them with our local students at the University of Canterbury starting in 1998. In 2002 we signed a contract with Addison-Wesley to provide the three ITSs and online quizzes on DatabasePlace, their Web portal (www.databaseplace.com). DatabasePlace was open in January 2003, serving two ITSs: SQL-Tutor, which teaches the SQL database query

language, and NORMIT, the data normalization tutor. ER-Tutor, an ITS that teaches database design using the Entity-Relationship data model, was added to the portal in January 2004.

Databases are ubiquitous in today's information systems. Our tutors are Web-enabled, and thus are classroom and platform independent. All three tutors are problem-solving environments, where the system presents problems to solve and offers adaptive problem-solving support and feedback.

In Section 2 we present a brief overview of the architecture and functionality of our tutors. The following three sections discuss SQL-Tutor, ER-Tutor and NORMIT. Section 5 presents some experiences with DatabasePlace, followed by conclusions in the last section.

2. Constraint-based tutors

Although ITSs have been proven to be effective in many domains, the number of ITSs used in real courses is still extremely small [7]. The typical architecture of our constraint-based tutors is given in Fig. 1. The tutors are developed in AllegroServe, an extensible Web server provided with Allegro Common Lisp. All student models are kept on the server. At the beginning of interaction, a student is required to enter his/her name, which is necessary in order to establish a session. The session manager requires the student modeller to retrieve the model for the student, if there is one, or to create a model for a new student. Each student action is sent to the session manager, to be linked to the appropriate session and stored in the student's log. The action is then sent to the pedagogical module (PM). If the submitted action is a solution to the current step, the PM sends it to the student modeller, which diagnoses the solution, updates the student model and sends the result of the diagnosis back to the PM, which generates feedback.

SQL-Tutor and NORMIT are Web-enabled tutors with a centralized architecture, with all tutoring functions performed on the server side. In these two domains, solutions produced by students are textual, and the amount of information to be sent to the server is small, so that the centralized architecture is appropriate. In ER-Tutor, students draw diagrams, and some tutoring functions related to drawing are performed on the client side. The tutoring functions are therefore distributed between the server and the Java applet, as described later.
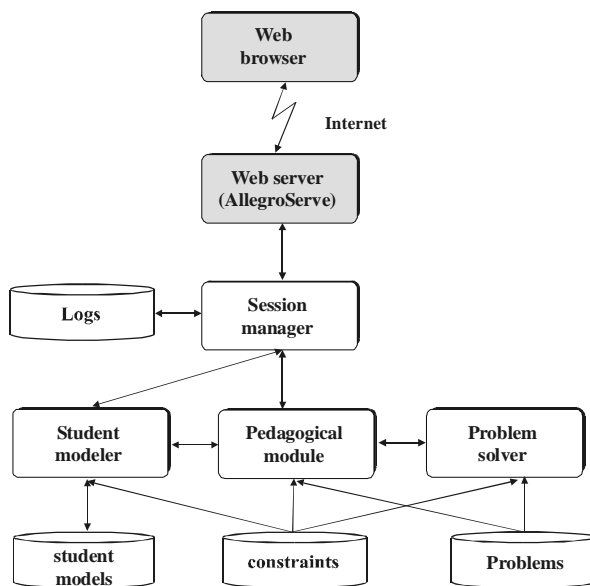

Fig. 1. The architecture of our tutors

Domain knowledge consists of a set of constraints. Constraint-Based Modeling (CBM) [8,10] is a student modeling approach that is not interested in the exact sequence of states in the problem space the student has traversed, but in what state he/she is in currently. As long as the student never reaches a state that is known to be wrong, they are free to perform whatever actions they please. The domain model is a collection of state descriptions of the form: *If <relevance condition> is true, then <satisfaction condition> had better also be true, otherwise something has gone wrong.* A simple example of a constraint is as follows: *If you are driving in New Zealand, you better be on the left side of the road.*

The knowledge base consists of constraints used for testing the student's solution for syntax errors and comparing it against the system's ideal solution to find semantic errors. The knowledge base enables the ITS to identify correct student solutions, no matter whether they are identical to the system's ideal solutions or

whether the student used an alternative way of solving the same problem. Constraints are problem-independent and modular, and therefore easy to evaluate. They are written in Lisp, and can contain built-in functions as well as domain-specific ones. For examples of constraints, please see [5,7,8,9,12]. If the satisfaction condition of a relevant constraint is met by the student solution, the solution is correct. In the opposite case, the student will be given feedback on errors.

One of the advantages of CBM over other student modeling approaches [6] is its independence from the problem-solving strategy employed by the student. CBM models students' evaluative, rather than generative knowledge and, therefore, does not attempt to induce the student's problem-solving strategy. CBM does not require an executable domain model, and is applicable in situations in which such a model would be difficult to construct (such as database design or SQL query generation). Furthermore, CBM eliminates the need for bug libraries, i.e. collections of typical errors made by students. On the contrary, CBM focuses on correct knowledge only. If a student performs an incorrect action, that action will violate some constraints. A violated constraint means that student's knowledge is incomplete/incorrect, and the system can respond by generating an appropriate feedback message.

The student modeller evaluates the student's solution against the knowledge base and updates the student model. The short-term student model consists of a list of violated and a list of satisfied constraints for the current attempt. The long-term model records the history of usage for each constraint. This information is used to select problems of appropriate complexity for the student, and to generate feedback.

All constraint-based tutors contain predefined database problems. ER-Tutor and SQL-Tutor also contain a pre-specified ideal solution for each problem, as there are no problem solvers for these two tutors. NORMIT, on the other hand, contains a problem solver, and is capable of solving both pre-specified problems and the problems entered by students.

The pedagogical module is the driving engine of the whole system. Its main tasks are to generate appropriate feedback messages for the student and to select new practice problems. PM individualizes these actions to each student based on their student model. Unlike ITSs based on model tracing [1,4], constraint-based tutors do not follow each student's solution step-by-step: a

student's solution is only evaluated once it is submitted, although the student may submit a partial solution to get ideas on how to progress.

3. SQL-Tutor

Students experience many problems when learning SQL. Some errors come from the burden of having to memorize database schemas; others come from misconceptions in the student's understanding of SQL and the relational data model. Furthermore, students find that it is not easy to learn SQL by working with a RDBMS, because error messages are very often hard to understand, and are limited to the syntax only.

The interface removes some of the cognitive load required for checking the low-level syntax, and enables the student to focus on query definition.

SQL-Tutor checks the student's solution by comparing it to the correct solution using domain knowledge represented in the form of about 700 constraints. The student may select problems in several ways: they may work their way through a series of problems for each database (ordered by their complexity), ask the system to select a problem on the basis of their student model, select a problem from a list, or select the type of problem they wish to work on, where the system then selects an individual problem of that type on the basis of their student model.
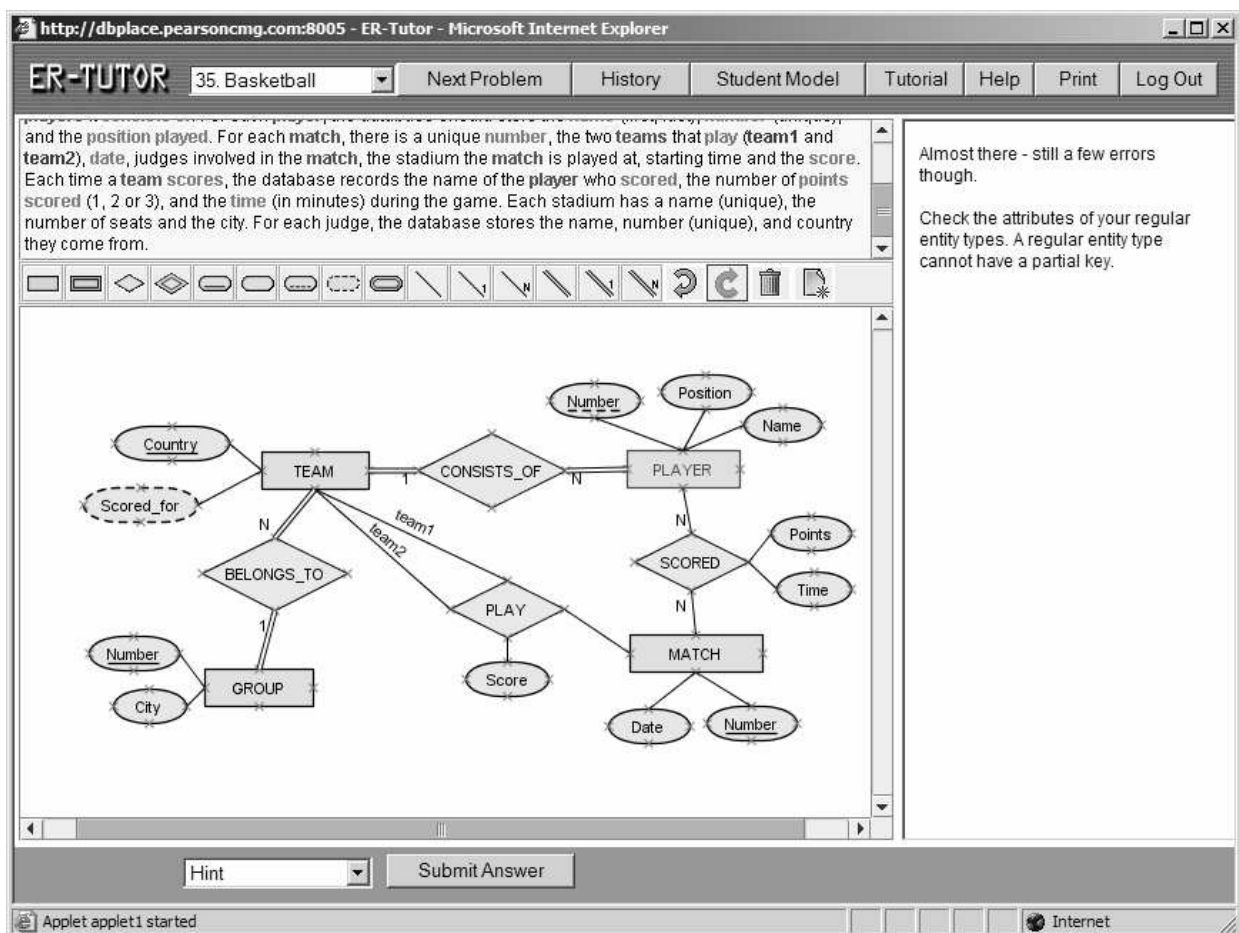

Fig. 2. The interface of ER-Tutor

The Web-enabled version of SQL-Tutor has been used in courses at the University of Canterbury since 1999. For a detailed discussion of the system, see [5,7,8]. The system contains several databases and a set of problems and their ideal solutions. The interface was designed to be robust, flexible, and easy to use. It reduces the memory load by displaying the database schema, the text of a problem, the basic structure of the query, and explanations of the elements of SQL.

4. ER-Tutor

Database design is a process of generating a database schema using a specific data model. The quality of conceptual schemas is of critical importance for database systems. Most database courses teach conceptual database design using the Entity-Relationship (ER) model, a high-level

data model [3]. Although the traditional method of learning ER modeling in a classroom environment may be sufficient as an introduction to database design, students cannot gain expertise by attending lectures only: like in other design tasks, extensive practise is necessary. ER-Tutor assists students in this task. The system is designed to complement classroom teaching, and therefore assumes that students are already familiar with the fundamentals of database theory. In ER-Tutor [9,12], students construct ER schemas that satisfy a given set of requirements. The system assists students during problem solving and guides them towards the correct solution by providing tailored feedback.

The system is designed for individual work. The student is given a textual description of the requirements of the database, and uses the ER modelling notation to construct an ER schema, as shown in Fig. 2. The interface consists of three main components. The top part contains the controls for the student to ask for a new problem, look at the history of the current session, explore their student model, ask for help or log out. The main component is the Java applet, which displays the text of the problem. It also provides an ER modeling workspace where students create ER diagrams. The feedback from the system is provided in the pane on the right. The ER diagram is constructed using the workspace integrated into the interface. Whenever a new object is created, the system asks for it to be named by highlighting a phrase from the problem text. This interface has two benefits: the student is forced to think about the requirements in terms of the original problem text, and it is also easier for the tutor to understand the semantics of the constructs in the student's diagram. Once the student has completed the problem or requires guidance from the system, the solution is evaluated. Depending on the results of the evaluation, the system may either congratulate the student or offer hints on their errors. The domain knowledge of ER-Tutor is represented as a set of 135 constraints, which is used for testing the student's solution (for syntax errors) and comparing it to the ideal solution.

5. NORMIT

Database normalization is the process of refining a relational database schema in order to ensure that all tables are of high quality [3]. Normalization is usually taught in introductory database courses in a series of lectures, and later practised on paper by looking at specific databases and applying the definitions. Database normalization is a procedural task: the student goes through a number of steps to analyze the quality of a database. We described the tasks NORMIT supports in detail elsewhere [9]. NORMIT requires the student to determine candidate keys, the closure of a set of attributes, prime attributes, simplify functional dependencies, determine normal forms, and, if necessary, decompose the table. The sequence is fixed: the student will only see a Web page corresponding to the current task. The student may submit a solution or request a new problem at any time. He/she may also review the history of the session, or examine their student model.

NORMIT currently contains over 80 problem-independent constraints that describe the basic principles of the domain. Some constraints check the syntax of the solution, while others check the semantics by comparing the student's solution to the ideal solution, generated by the problem solver. In order to identify constraints, we studied material in textbooks, such as [3], and also used our own experience in teaching database normalization.

6. DatabasePlace

The number of DatabasePlace users has been increasing steadily, as illustrated in Fig. 3. Students get access to the portal by buying a database book published by Addison-Wesley, or by obtaining access directly from the Web. The contract with the publisher does not allow for collecting general information about the users or their background knowledge prior to using the ITSs, but we do have access to session logs. Although we do not know where the users come from, it is evident from the figure that most of the students come from the northern hemisphere; there are fewer new users during July-August period. ER-Tutor was available on the portal a year later than the other two tutors, but it is equally popular. NORMIT seems to attract the least number of users, which is not surprising, taking into account the highly theoretical nature of its instructional area.

We performed numerous evaluation studies on these three ITSs with local students at the University of Canterbury, the results of which show that they increase students' knowledge significantly [7,8,9,12]. The ITSs are especially effective for less able students, although we have proofs that there are also beneficial for more

advanced students. Subjective information shows that students appreciate working with the tutors, as they are available at any time and from any place, and especially praise the feedback provided. Since we have no knowledge of backgrounds of students using the same systems on DatabasePlace, it is interesting to see whether the same effects are achieved on the portal.

We performed various analyses of student data collected on the portal. DatabasePlace students prefer different problems than Canterbury students. For example, Canterbury students most often select problems from two databases offered in SQL-Tutor (MOVIE and COMPANY), while DatabasePlace students prefer other databases. These superficial differences can be explained easily: Canterbury students use SQL-Tutor in the labs, as a complement to lectures. In the course taught by the author, many examples used in lectures come from these two databases. DatabasePlace students, on the other hand, use SQL-Tutor (most likely) completely independently from the courses they are enrolled in, and select databases based on their own preferences.
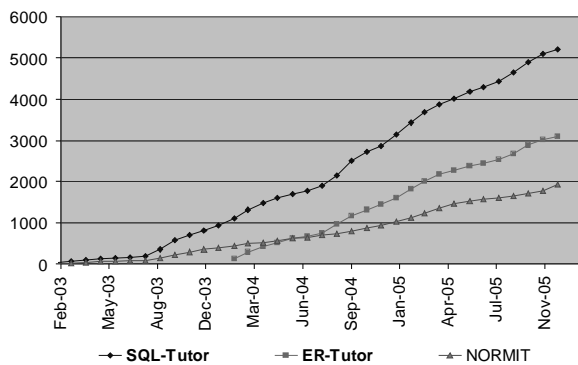

Fig. 3. The number of DatabasePlace users

A more interesting analysis looks at the completion rates. On average, 3-15% of Canterbury students log on to the ITSs without even making any attempts at solving problems. The percentage of students behaving the same way in DatabasePlace is higher, and ranges from 30% to 45%. The percentage of students who complete no problems for Canterbury students ranges from 3% to 12% (depending on the system), while for DatabasePlace students this range is much wider (12-40%). We believe this illustrates the effect of having no human teacher in the loop: Canterbury students are told in lectures that the tutoring systems are useful for practice, and that they may help students learn

better, while no such reinforcement is there for DatabasePlace students.
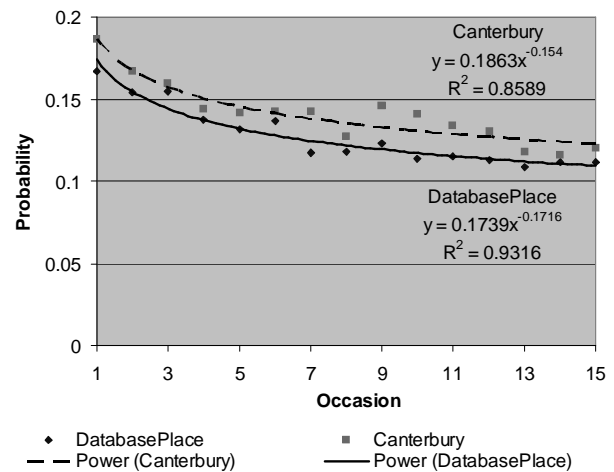

Fig. 4. Learning curves

The most important analysis is whether the two groups of students learn equally well. Figure 4 shows the learning curves for students using NORMIT. We compared two groups of students: the Canterbury group included all students participating in a study performed in 2004, while the DatabasePlace group consists of all students using NORMIT on DatabasePlace. To produce the learning curve, we calculate the probability of violating a constraint on its nth occasion of being relevant. This probability is then averaged over all constraints and over all students. Fig. 4 shows the raw data points and also the fitted power curves. It can be seen that both power curves represent very good approximations of the data sets, with the R^2 fits of 0.86 and 0.93 for the Canterbury and DatabasePlace groups respectively. A good fit to the power curve is widely accepted in the ITS area as a measure of the psychological appropriateness of the used knowledge representation formalism; in our case, these graphs show that students indeed do acquire knowledge in the area as represented in the system (i.e. the students learn constraints). The initial probability of errors is slightly higher for Canterbury students (0.19) than for Database Place students (0.17), but the difference is not significant. The learning rates (i.e. the exponents of the power curves) are comparable, meaning that both groups learn equally well. The slightly higher R^2 for the DatabasePlace group is the statistical effect of a much larger size of the group; there were less than 50 Canterbury students, compared to almost two thousand students using NORMIT on DatabasePlace.

7. Conclusions

We presented three of our constraint-based tutors for the database area, which are used with local students at the University of Canterbury, and also at a Web portal with worldwide students. The DatabasePlace Web portal has been active for more than three years now, and the three ITSs available on it have been used by several thousand students. At Canterbury, we have conducted multiple evaluation studies since 1998, but with much smaller populations of students. The analyses we performed on student logs collected both locally and from DatabasePlace show that both groups of students learn equally effectively using these systems, although there are differences in attrition rates and problem completion rates. We believe that it is beneficial to have the teacher involved in the process, as is the case with Canterbury students, which increases student participation and motivation. It is encouraging, though, to see that students' learning is not affected by not having the teacher actively involved, as students learn equally well on the DatabasePlace portal.

Our experience shows that ITSs have reached the maturity level at which they provide a successful and widely accessible platform for learning. We believe that ITSs will become much more frequent in classrooms and also much more widely used in e-learning courses. The biggest barrier at the moment is the difficulty of developing new ITSs, as they require not only domain expertise, but also expertise in software development, psychology and education. Our current work is focused on developing ASPIRE, a Web-enabled authoring system for constraint-based tutors. ASPIRE will support teachers in developing ITSs for their students, without requiring programming expertise. Our authoring system will provide all required functionality, and support the author in the process of specifying the domain model, which is the most difficult and time-consuming task in ITS development. Domain models will be induced using machine-learning techniques, from the domain information and examples of solved problems supplied by teachers.

8. References

[1] Anderson JR, Corbett AT, Koedinger KR, Pelletier R. Cognitive Tutors: Lessons Learned. The Journal of the Learning Sciences 1995; 4(2): 167-207.

[2] Bloom B.S. The 2-sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher 1984; 13: 4-16.

[3] Elmasri R, Navathe S.B. Fundamentals of Database Systems. Addison Wesley, 2004.

[4] Koedinger, K. R., Anderson, J. R., Hadley, W. H. & Mark, M. A. Intelligent tutoring goes to school in the big city. Artificial Intelligence in Education 1997; 8(1): 30-43.

[5] Mitrovic A. Experiences in Implementing Constraint-Based Modelling in SQL-Tutor. In: Goettl BP, Halff HM, Redfield CL, Shute VJ, editors. Proceedings of the 4th International Conference on Intelligent Tutoring Systems, 1998, p. 414-423.

[6] Mitrovic A, Koedinger K, Martin B. A Comparative Analysis of Cognitive Tutoring and Constraint-Based Modelling. In: Brusilovsky P, Corbett A, de Rosis F, editors. Proceedings of the 9th International Conference on User Modeling, Springer-Verlag, LNAI 2702, 2003, p. 313-322.

[7] Mitrovic A, Martin B, Mayo M. Using Evaluation to Shape ITS Design: Results and Experiences with SQL-Tutor. User Modeling and User-Adapted Interaction 2002; 12(2-3): 243-279.

[8] Mitrovic A, Ohlsson S. Evaluation of a Constraint-Based Tutor for a Database Language. Artificial Intelligence in Education 1999; 10(3-4): 238-256.

[9] Mitrovic A, Suraweera P., Martin B, Weerasinghe A. DB-suite: Experiences with Three Intelligent, Web-based Database Tutors. Interactive Learning Research 2004; 15(4): 409-432.

[10] Ohlsson S. Constraint-based Student Modelling. In Proc. of Student Modelling: the Key to Individualized Knowledge-based Instruction, Springer-Verlag, Berlin, pp. 167-189, 1994.

[11] Self JA. Theoretical foundations for intelligent tutoring systems. Artificial Intelligence in Education 1990; 1(4): 3-14.

[12] Suraweera P, Mitrovic A. An Intelligent Tutoring System for Entity-Relationship Modelling. Artificial Intelligence in Education 2004; 14(3-4): 375-417.