

Constraint-Based Tutors

Antonija Mitrovic and the ICTG team

Intelligent Computer Tutoring Group
Department of Computer Science, University of Canterbury
Private Bag 4800, Christchurch, New Zealand
Tanja.mitrovic@canterbury.ac.nz

1 Introduction

Intelligent Computer tutoring Group (ICTG) has developed several constraint-based tutors. We adopted and enhanced Constraint-Based Modeling (CBM), a student modeling approach proposed by Ohlsson (1994). In this interactive demo, we will present our three database tutors, which have also been commercialized on the Addison-Wesley's DatabasePlace Web portal (<http://www.aw-bc.com/databaseplace/>). All our ITSs are Web-enabled, and their typical architecture is shown in Figure 1. The session manager is responsible for taking care of student sessions, and logging their actions. The student communicates with the ITS through an interface served in a Web browser. Student models are generated and maintained by a student modeller. The student's solution is diagnosed by matching it to the constraints describing the domain principles. The pedagogical module determines the timing and content of pedagogical actions. The optional components involve the problem solver, the group modeller and modules supporting open student models and meta-cognitive skills. We have performed more than 30 studies in real classrooms since 1998. Our database tutors have proved to be very effective in supporting deep learning, and are well liked by students.

2 SQL-Tutor

SQL-Tutor is a constraint-based tutor that helps university-level students to learn SQL. The system contains definitions of several databases, and a set of problems and the ideal solutions to them. SQL-

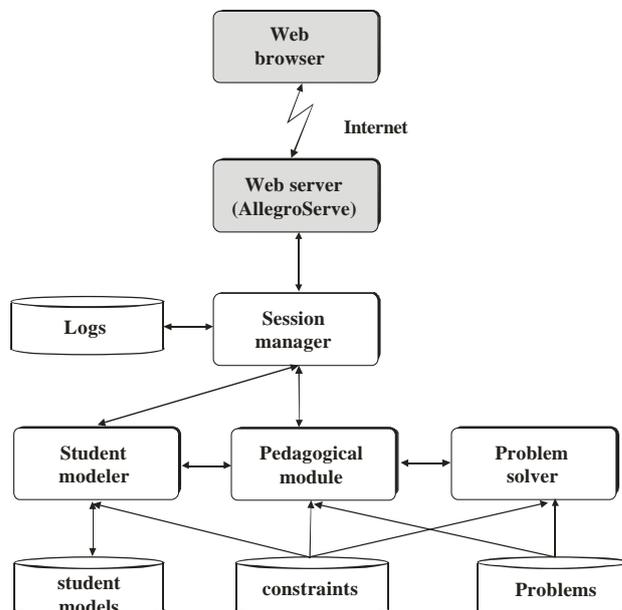


Fig. 1. Architecture of constraint-based ITSs

Tutor contains no domain module, and is not capable of solving the problems on its own. In order to check the correctness of the student's solution, SQL-Tutor compares it to the correct solution, using domain knowledge represented in the form 700 constraints. At the beginning of a session, SQL-Tutor selects a problem for the student to work on. When the student submits a solution, the pedagogical module sends it to the student modeller, which analyzes the solution, identifies mistakes (if there are any) and updates the student model appropriately. On the basis of the student model, the pedagogical module generates an appropriate pedagogical action (i.e. feedback). When the current problem is solved, or the

student requires a new problem to work on, the pedagogical module selects an appropriate problem on the basis of the student model.

The interface has been designed to be robust, flexible, and easy to use. It reduces the memory load by displaying the database schema and the problem text, by providing the basic structure of the query, and also by providing explanations of the elements of SQL. The main page is divided into three areas. The upper part displays the text of the problem being solved and students can remind themselves easily of the elements requested in queries. The middle part contains the clauses of the SELECT statement, thus visualizing the goal structure. Students need not remember the exact keywords used and the relative order of clauses. The lowest part displays the schema of the chosen database. Schema visualization is very important; all database users are painfully aware of the constant need to remember table and attribute names and the corresponding semantics as well. Students can get the descriptions of databases, tables and/or attributes. The motivation here is to remove from the student some of the cognitive load required for checking the low-level syntax, and to enable the student to focus on higher-level, query definition problems.

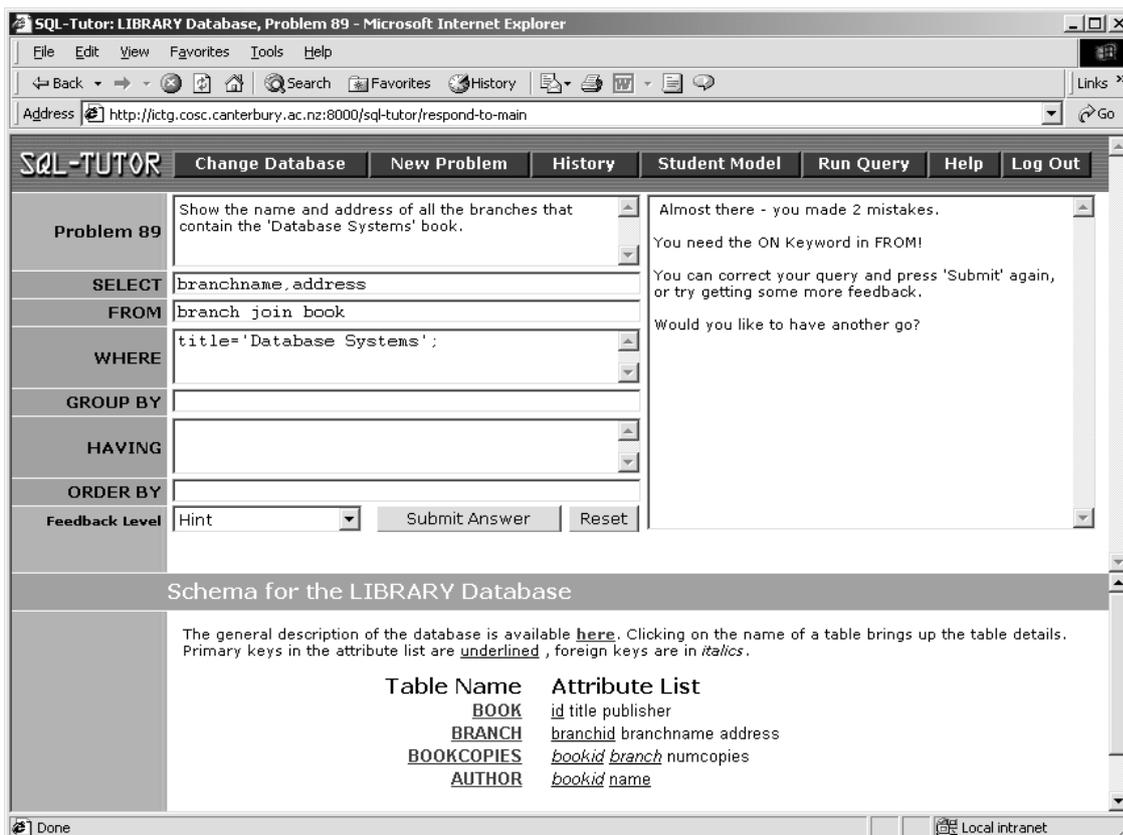


Fig. 2. The interface of SQL-Tutor

Students have several ways of selecting problems in SQL-Tutor. They may work their way through a series of problems for each database, ordered by their complexity, or the system to select the problem, based of the student model.

SQL-Tutor was evaluated in 11 studies since 1998, which proved the system's effectiveness (Mitrovic et al, 1999, 2004, 2007). All the studies showed that CBM has sound psychological foundations and that students acquire constraints at a high rate.

3 EER-Tutor

Database modelling is a complicated task, which requires significant amounts of practice to achieve expertise. Conceptual database modelling is traditionally taught in a classroom environment where concepts and solutions to typical databases are explained. Our experiences point to the need of providing students with individualized feedback, as students' solutions differ enormously between each other. We have developed EER-Tutor, an ITS that teaches conceptual database modelling using the EER model, a popular high-level data model (Suraweera & Mitrovic, 2004; Zakharov et al., 2005). The top section of the interface displays the problem text to the student, and provides the

controls for selecting problems, viewing the student model, printing the diagram etc. The main part of the interface is the drawing workspace, containing the tools for developing EER diagrams. Students can drag and drop constructs onto their workspace to construct EER schemas. The feedback window displays textual pedagogical messages. As in other constraint-based tutors, EER-Tutor provides several levels of feedback. The first level of feedback simply indicates whether the submitted solution is correct or not. The “Error flag” indicates the type of construct (e.g. entity, relationship, etc.) that is incorrect. “Hint” and “detailed hint” offer instructions on the most important error. “Hint” is a general message, whereas “detailed hint” provides a more specific message. Feedback on all violated constraints is displayed at the “all errors” level. The complete solution is presented as an image in the final level.

We implemented several version of EER-Tutor; in addition to the original version, there is a version supporting self-explanation, and a version with the animated, affect-sensitive pedagogical agent (Zakharov et al., 2008).

3 NORMIT

NORMIT teaches data normalization in relational databases. As data normalization is a procedural task, it was possible to implement a problem-solver, and therefore in this system we do not store pre-specified ideal solutions. The interesting feature of this tutor is the procedural nature of the task, which required constraints different from the ones we had in previous tutors. Database normalization is a procedural task: the student goes through a number of steps to analyze the quality of a database. 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 step of the procedure. 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. There is also a version of NORMIT which supports self-explanation, in which the student need to justify his/her actions by providing reasons for them, and also by providing definitions of the underlying domain concepts. Evaluation studies performed show that NORMIT helps student acquire problem-solving skills, and that the self-explanation support also results in increased declarative knowledge.

References

1. Mitrovic, A., Martin, B., Suraweera, P. Intelligent tutors for all: Constraint-based modeling methodology, systems and authoring. *IEEE Intelligent Systems*, special issue on Intelligent Educational Systems, vol. 22, no. 4, pp. 38-45, July/August 2007.
2. Mitrovic, A., Suraweera, P., Martin, B. and Weerasinghe, A. (2004) DB-suite: Experiences with Three Intelligent, Web-based Database Tutors. *Journal of Interactive Learning Research*, 15(4), pp. 409-432, 2004.
3. Suraweera, P. and Mitrovic, A., An Intelligent Tutoring System for Entity Relationship Modelling. *Int. J. Artificial Intelligence in Education*, 14 (3-4), 375-417, 2004.
4. Mitrovic, A., Ohlsson, S.: Evaluation of a Constraint-based Tutor for a Database Language. *Int. J. on Artificial Intelligence in Education*, 10(3-4), 238-256, 1999.
5. Ohlsson, S.: Constraint-based student modeling. In: Greer, J.E., McCalla, G (eds): *Student modeling: the key to individualized knowledge-based instruction*, 167-189, 1994.
6. Zakharov, K., Mitrovic, A., Ohlsson, S., Feedback Micro-engineering in EER-Tutor. In: C-K Looi, G. McCalla, B. Bredeweg, J. Breuker (eds) *Proc. Artificial Intelligence in Education AIED 2005*, IOS Press, pp. 718-725, 2005.
7. Zakharov, K., Mitrovic, A., Johnston, L. Towards Emotionally-Intelligent Pedagogical Agents. Accepted for ITS 2008.