Int J Artif Intell Educ (2016) 26:150–159
DOI 10.1007/s40593-015-0049-9

ARTICLE

Implementing CBM: SQL-Tutor After Fifteen Years

Antonija Mitrovic[1] · **Stellan Ohlsson**[2]

Published online: 28 May 2015

Abstract SQL-Tutor is the first constraint-based tutor. The initial conference papers about the system were published in 1998 (Mitrovic 1998a, 1998b, 1998c), with an IJAIED paper published in 1999 (Mitrovic and Ohlsson, *International Journal Artificial Intelligence in Education, 10*(3–4), 238–256, 1999). We published another IJAIED paper in 2003, focussed on the Web-enabled version of the same system (Mitrovic, *Artificial Intelligence in Education, 13*(2–4), 173–197, 2003). In this paper, we discuss the reasons for developing the system, our experiences with the early versions, and also provide a history of later projects involving SQL-Tutor.

Keywords SQL-Tutor · Teaching querying · Ill-defined task

Introduction

Constraint-based modelling (CBM) was proposed by the second author as a way to overcome limitations of other student modeling approaches existing at the time (Ohlsson 1992). As discussed in another paper in this volume (Ohlsson 2015), CBM provided a way to eliminate the need for bug libraries. A constraint-based tutor only needs a set of constraints that describe features of correct solutions. In addition, CBM allowed us to expand the types of instructional tasks that can be taught by an intelligent tutor. CBM does not require a runnable domain module (i.e., a problem solver), so it can be applied to open-ended tasks for which there is no algorithmic solution or executable expert model.

The original paper on CBM attracted a lot of attention, as illustrated by the early citations. However, at the time of its publication, and for several years afterwards, there

✉ Antonija Mitrovic
tanja.mitrovic@canterbury.ac.nz

Stellan Ohlsson
stellan@uic.edu

[1] Intelligent Computer Tutoring Group, Department of Computer Science and Software Engineering, University of Canterbury, Christchurch, New Zealand

[2] Department of Psychology, University of Illinois at Chicago, Chicago, IL, USA

was no system based on the proposed approach. The first author of this paper completed her Ph.D. in 1994, in which she proposed another student modelling approach particularly suited to procedural tasks, implemented in a system called INSTRUCT. It combined advantages of model tracing (Anderson et al. 1990) with those of reconstructive modelling as implemented in the ACM system (Ohlsson and Langley 1988). In the process of revising a paper about INSTRUCT (Mitrovic et al. 1996), Mitrovic learnt about CBM and found it extremely exciting! Ohlsson proposed solutions to some of the problems with previous student modelling approaches (including the one used in INSTRUCT). The motivation for implementing the SQL-Tutor was to investigate whether CBM was as promising as it seemed.

The choice of instructional task was influenced by the Computer Science courses Mitrovic was teaching. She observed over the years that students found the SQL database query language very challenging: although the language itself is well-defined in terms of its grammar, the task of writing queries in SQL is demanding. SQL queries are posed in English; they require a good understanding of the database which serves as the context, and are often ambiguous, requiring common-sense and background knowledge. Additionally, the student needs to be familiar with the relational data model and the Database Management System (DBMS) in which they practise composing SQL queries. DBMSs can be difficult to learn, and they typically provide cryptic error messages in response to syntax errors, but they are not capable of dealing with semantic errors. Finally, writing SQL queries is a design task, a simplified version of programming, and there is no algorithm that students can apply to convert the natural language description of a query into an SQL *Select* statement (Mitrovic and Weerasinghe 2009). Because SQL has a lot of redundancy built into it, there are often several correct solutions for one and the same problem. For all these reasons, students find it hard to master the art of writing SQL queries.

Consequently, SQL was an ideal domain in which to develop the first constraint-based tutor: a demanding instructional task, included in real courses with real learners who needed to learn SQL. Mitrovic started developing SQL-Tutor at the end of 1995. In 1998, while on sabbatical, she visited Ohlsson at the University of Illinois at Chicago and demonstrated SQL-Tutor. That was the start of our very productive collaboration, which we still enjoy wholeheartedly! Numerous later constraint-based tutoring systems have been developed within the Intelligent Computer Tutoring Group (ICTG[1]), led by Mitrovic at the University of Canterbury in Christchurch, New Zealand.

We start by briefly discussing the two highly cited IJAIED papers on SQL-Tutor. The subsequent section presents a brief history of later projects involving SQL-Tutor, and our future plans. We conclude with presenting the lectures learnt from SQL-Tutor.

Early Versions of SQL-Tutor

SQL-Tutor was designed as a complement to database courses. It assumes that the student has already acquired some knowledge via lectures, labs and demonstrations. The system provides numerous problem-solving opportunities in the context of various databases. Students can freely choose among the latter. The system covers only the

[1] www.ictg.canterbury.ac.nz

SQL Select statement (more precisely, a subset of language constructs used in the Select statement). The system is focused on querying, because queries cause most of the student misconceptions. Additionally, many concepts used in querying are directly relevant to other SQL statements and even to other relational database languages. SQL-Tutor was developed in Allegro Common Lisp,[2] with the first version being implemented on Solaris workstations, and a later version as a stand-alone MS Windows application.

Mitrovic's team faced several challenges and issues in the design and development of SQL-Tutor. These included the nature of constraints, the application of constraints, long-term modeling, interface design, pedagogical approach, and evaluation methodology.

Nature of constraints Because SQL Tutor was the first constraint-based tutor, the first problem was to operationalize the theoretical definition of constraints. This required several conceptual developments. Mitrovic classified constraints into *syntax* and *semantic* constraints: The former focused on the syntactical correctness of the student's solution, while the semantic constraints made sure that the student's solution was correct for the problem at hand (Mitrovic 1997; Mitrovic 1998c). Both types of constraints were problem-independent; none of the constraints included any problem-specific elements. This feature makes it easy to add new practice problems to SQL-Tutor (and other constraint-based tutors). The system builder only needs to provide the text of the problem and one correct solution. In the case of a new database, the author needs only provide the description of the database. Because most problems have several correct solutions, semantic constraints are needed to check for all correct ways of specifying the elements of a query. In addition, Mitrovic's work brought into clearer focus the distinction between *state constraints* and *path constraints*. The former refer to a single problem state, while the latter refer to a sequence of events. Although Ohlsson's original '92 formulation emphasized state constraints, in practice path constraints are needed to catch all student errors.

Application of constraints The next challenge was related to implementation of short-term modelling. To analyse a student's solution, it is first necessary to match the relevance conditions of all constraints to find those that are relevant to the solution. In the next step, the satisfaction conditions of relevant constraints are used to determine whether they are violated or satisfied. At the time when the first version of SQL-Tutor was developed, sequential matching of constraints was not feasible due to the size of the knowledge base,[3] so Mitrovic implemented a modification of the RETE pattern matcher often used in rule-based AI systems (Forgy 1982) to speed up matching. This resulted in relevance and satisfaction networks, providing information about relevant and satisfied/violated constraints (Mitrovic 1997).

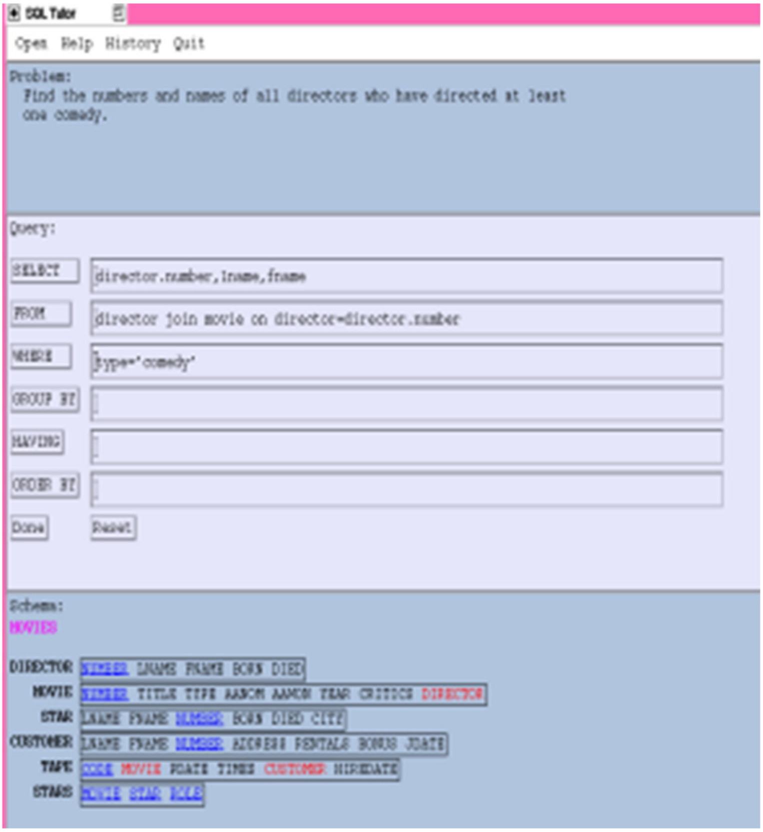
Long-term modeling The next step was to decide on the long-term modelling approach. CBM as proposed by Ohlsson was focused on short-term modelling (or error diagnosis), but ITSs need to track the progress of students over longer time periods.

[2] Franz.com

[3] The version of SQL-Tutor used in the first evaluation study in 1998 had 400 constraints, while the current version has more than 700 constraints.

Because constraints are rather different in character from other types of knowledge representations (networks, rules, etc.), these issues had to be thought through from scratch. The first version of the student model in SQL-Tutor was an overlay on the constraint base (Mitrovic 1997). This student model tracks student learning by storing the history of each constraint (and some additional information). This information can be analyzed in many ways. One simple approach is to identify the set of situations within a recent time frame in which some constraint C was relevant, and calculate the proportion of those situations in which it was also satisfied.

Interface design Another challenge, which is faced by all ITSs, was to design the interface, which should support a natural way of solving problems in the domain and at the same time provide problem-solving support to the student. Figure 1 shows the interface used in the first evaluation study performed in 1998 (Mitrovic 1997). The interface of SQL-Tutor reduces the student's working memory load by displaying the database schema (the bottom part) and problem text (at the top), by providing the basic structure of the query and also by providing explanations of various elements of SQL. In addition, SQL-Tutor provides feedback on student solutions at several levels of specificity, ranging from telling the student whether the solution is correct, letting the student know which part of the solution is wrong (*error flag*), providing hint-level



Problem:
Find the numbers and names of all directors who have directed at least one comedy.

Query:

SELECT director.number,lname,fname

FROM director join movie on director=director.number

WHERE type='comedy'

GROUP BY

HAVING

ORDER BY

Schema:
MOVIES

DIRECTOR NUMBER LNAME FNAME BORN DIED
MOVIE NUMBER TITLE TYPE AANOM AAWON YEAR CRITICS DIRECTOR
STAR LNAME FNAME NUMBER BORN DIED CITY
CUSTOMER LNAME FNAME NUMBER ADDRESS RENTALS BONUS JDATE
TAPE CODE MOVIE PDATE TIMES CUSTOMER HIREDATE
STARS MOVIE STAR ROLE

Fig. 1 Screenshot of the Solaris version of SQL-Tutor

messages about one or all violated constraints, to providing partial or even a full solution.

Pedagogical approach Other challenges included deciding on the pedagogical approach to be used, and how to exploit the constraint formalism to support that approach. This included feedback (both content and timing) and problem selection strategies. SQL-Tutor allows the student to select a database as a context for problem solving. After that, the student can select a problem, or ask the system to select the problem on the basis of the student model (Mitrovic 1997). In the 1998 study, the system selected problems based on constraints that a student had difficulties with, and/or constraints that had not been relevant for any of his or her problems or solutions so far (Mitrovic and Ohlsson 1999). In this way, the constraint base served as a tool for moving the student through the curriculum in a way that is sensitive to what he or she has or has not yet learned.

Evaluation methodology The 1999 paper presented the findings from the first evaluation study conducted in 1998. Although the study was uncontrolled, with volunteers using the system, it confirmed that constraints are an appropriate formalism for representing the task knowledge, and also that students learned from the system and found it useful and easy to use. In the course of analyzing the data, we had to figure out how to relate the data collected by a CBM system to classical representations of learning, such as learning curves. We displayed the results by plotting the probability that a constraint is violated as a function of the number of situations in which that constraint was relevant, averaged over constraints and students. This type of plot turned out to yield smooth learning curves that followed the power law of learning.

In short, the translation from the idea of constraint-based student modeling to a working tutoring system presented the Mitrovic team with multiple issues and challenges. Some issues had to be thought through or re-thought from scratch because the constraint formalism that is at the center of the CBM approach is different in character and operates differently from other, more traditional knowledge representation. In the end, all the issues were resolved and SQL-Tutor was, and remains, a successful system. It also remains true that the constraint formalism is difficult to grasp, and it is the aspect of CBM that is most often misunderstood.[4]

What Happened Later?

In the years after the 1999 paper, SQL-Tutor was improved, expanded, and used as a research platform in multiple ways. One important aspect of these developments was re-implementing the system to make it more universally accessible. ICTG developed a MS Windows version, which was downloaded 1946 times from May 1999 to January 2001. Mitrovic (2003) described the web-enhanced version of SQL-Tutor, which has

[4] An example of such a misconception is the mistaken idea that constraint-based ITSs require something called "buggy constraints" (Zinn 2014; Kodaganallur et al. 2005). However, the point of constraints is to encode correct knowledge about the target task (Mitrovic and Ohlsson 2006).

been used in studies since 1999. A web-enabled version was initially developed using CL-HTTP, and later the Allegro Serve Web server. Since 2003, SQL-Tutor has also been available on the Addison-Wesley's DatabasePlace[5] Web portal. The pedagogical effectiveness of SQL-Tutor has been confirmed in 16 different studies, and the system has been in regular use in the University of Canterbury database courses, as well as in courses at other universities worldwide.

Several studies focused on the content of the system's feedback messages. An early study (Mitrovic and Suraweera 2000) compared the effectiveness of feedback provided in the textual form to the identical feedback being presented by an animated pedagogical agent, showing the persona effect. We also investigated the effectiveness of various levels of feedback (Mitrovic and Martin 2000); the results showed that feedback presenting information about the violated domain principles (e.g., *Hint* and *All errors*) was superior to other feedback levels, and resulted in faster and more effective learning. Originally SQL-Tutor only provided negative feedback (i.e., feedback on errors); when we added positive feedback, students were able to complete the same learning tasks in only half of the time while achieving the same learning improvement (Mitrovic et al. 2013).

Problem selection has also been investigated in several studies. As stated previously, early versions of SQL-Tutor selected problems based on constraints that students found difficult (i.e., frequently violated), or that they had not yet encountered (the default strategy). We experimented with other representations of the student's long-term knowledge based on Bayesian networks, and used them to select problems adaptively (Mayo and Mitrovic 2001). A small-scale study of that approach showed that the Bayesian problem-selection strategy resulted in problems that were better tailored to students' knowledge, in terms of the coverage of domain concepts and problem-solving effort required, compared to the default strategy (Mitrovic et al. 2002). Additionally, we investigated using artificial neural networks for problem selection (Wang and Mitrovic 2002); we trained a simple feed-forward network to predict the number of errors a student will make, and used such predictions to select the next problem. We also developed an approach to generating problems automatically, from the domain model; the results showed that such problems doubled the learning rate compared to the manually defined problems (Martin and Mitrovic 2002).

We have developed several versions of Open Student Models (OSM) as a vehicle to support students in self-assessment and reflection (Mitrovic and Martin 2007). The results showed that the students became more aware of deficiencies in their knowledge, and they made better decisions during learning. In addition to pure problem solving, we also explored the effect of learning from examples; the results showed that students learn significantly more from alterative presentation of examples and problems than when only examples are presented, and that they also acquire significantly more conceptual knowledge compared to problem-solving alone (Shareghi Najar and Mitrovic 2013). Furthermore, adaptive selection of examples and problems resulted in significantly improved learning compared to a fixed sequence of example/problem pairs (Shareghi Najar et al. 2014).

The work on SQL-Tutor opened the way for other projects. Some of these were designed to demonstrate that CBM is an effective way of modelling domain and student

[5] http://www.aw-bc.com/databaseplace/

knowledge in a variety of instructional tasks. We developed successful constraint-based tutors for other design tasks, such as database design (Suraweera and Mitrovic 2004; Zakharov et al. 2005), UML class diagrams (Baghaei and Mitrovic 2006) and Java programming (Holland et al. 2009). We also developed constraint-based tutors for procedural tasks, such as data normalization (Mitrovic 2005). Having proved that CBM can be used in a wide range of tasks, we turned to other interesting research problems. These include providing feedback on collaboration for pairs of students (Baghaei et al. 2007) and supporting students' affective state (Zakharov et al. 2008). SQL-Tutor also provided motivation and foundations for our work on authoring support for constraint-based tutors (Martin et al. 2008; Mitrovic et al. 2009). The interested reader is referred to (Mitrovic 2012) for coverage of other ICTG projects than those directly related to SQL-Tutor.

The 1999 and 2003 IJAIED papers on SQL-Tutor also contributed to popularization of CBM. Since those early days, CBM has been used by many researchers in addition to those coming from ICTG – see e.g., (Billingsley et al. 2004; Billingsley and Robinson 2005; Rosatelli and Self 2004; Riccucci et al. 2005; Petry and Rosatelli 2006; Menzel 2006; Mills and Dalgarno 2007; Siddappa and Manjunath 2008, Oh et al. 2009; Faria et al. 2009; Galvez et al. 2009a, b; Le 2006; Le et al. 2009; Roll et al. 2010; Poitras and Poitras 2013; Zinn 2014).

Our work on SQL-Tutor and other constraint-based tutors has demonstrated the wide applicability of CBM; it can be applied to both well-defined and ill-defined domains and tasks (Mitrovic and Weerasinghe 2009). In the future, we will continue developing constraint-based tutors to provide further evidence of the strengths of this methodology. Furthermore, we believe that, in order to reach the effectiveness of expert human tutors, ITSs need to support multiple learning mechanisms (Ohlsson 2008). Our future research plans are organized around this belief; we are currently enhancing SQL-Tutor to support new instructional strategies in addition to problem solving and learning from examples.

Reflections

The development of SQL-Tutor and the entire line of CBM-based tutoring systems is a prototypical example of a successful research and development process, with a heavy emphasis on "and." Without basic inquiry into the nature of cognitive skills and how they might be acquired, the hypothesis of constraint-based skill specialization might never have been discovered. On the other hand, without the willingness of technologists to implement systems, the problems and weaknesses of bug library based tutoring systems might not have been seen clearly enough to warrant searching for an alternative approach. In the end, theoretical concepts and efforts at implementation came together because the two authors of this paper reached out to each other across the theory-technology gap. It is often said that such bridges are necessary and productive, but institutional disincentives to gap-bridging all too often get in the way. The first lesson of SQL-Tutor in particular and the CBM approach in general for ITS researchers is thus simple to state but sometimes hard to execute: *Keep talking*.

The entire process since the 1999 paper follows a radiating pattern. SQL-Tutor served as a platform for research on a wide variety of tutoring related problems. In addition, it served as a paradigm and template for other tutoring systems operating in other instructional domains. These latter systems have in turn served as platforms for yet other studies and projects. Looking down the history of technology, we can see other examples of this *core + variants* pattern: Think of the Wright Brothers' first biplane and all the different types of airplanes that followed; the first vaccine and the many that followed; the first sky scraper and the many variants that now stand tall all over the world. In technology, exploring a design space by building gizmos that are variants of existing, successful gizmos is a standard operating procedure. The field of ITS research might benefit from keeping this pattern in mind.

Acknowledgments The work reported here could not have been done without the wonderful bunch of students and colleagues at ICTG. Thank you all for the discussions and friendship over the years – we have been privileged to work with you.

References

Anderson, J. R., Boyle, C. F., Corbett, A. T., & Lewis, M. W. (1990). Cognitive modeling and intelligent tutoring. *Artificial Intelligence, 42*, 7–49.

Baghaei, N., & Mitrovic, A. (2006) *A Constraint-based collaborative environment for learning UML class diagrams*. In M. Ikeda, K. Ashley, T.-W. Chan (Eds.) Proc. 8th Int. Conf. Intelligent Tutoring Systems (pp. 176–186).

Baghaei, N., Mitrovic, A., & Irwin, W. (2007). Supporting collaborative learning and problem-solving in a constraint-based CSCL environment for UML class diagrams. *Computer-Supported Collaborative Learning, 2*(2–3), 159–190.

Billingsley, W. & Robinson, P. (2005) *Towards an intelligent online book for discrete mathematics*. Proc. Int. Conf. Active Media Technology, (pp. 291–296).

Billingsley, W., Robinson, P., Ashdown, M., & Hanson, C. (2004) *Intelligent tutoring and supervised problem solving in the browser*. Proc. Int. Conf. WWW/Internet (pp. 806–811).

Faria, L., Silva, A., Vale, Z., & Marques, A. (2009). Training control centers' operators in incident diagnosis and power restoration using intelligent tutoring systems. *IEEE Transactions on Learning Technologies, 2*(2), 135–147.

Forgy, C. L. (1982). Rete: a fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence, 19*, 17–37.

Galvez, J., Guzman, E., Conejo, R. & Millan, E. (2009a) *Student knowledge diagnosis using item response theory and constraint-based modeling*. In V. Dimitrova, R. Mizoguchi, B. du Boulay, A. Graesser (eds.) Proc 14th Int. Conf. Artificial Intelligence in Education, (pp. 291–298).

Galvez, J., Guzman, E., & Conejo, R. (2009b). A blended e-learning experience in a course of object oriented programming fundamentals'. *Knowledge-Based Systems, 22*(4), 279–286.

Holland, J., Mitrovic, A., & Martin, B. (2009) *A Constraint-based tutor for java*. In S.C. Kong, et al. (Eds.) Proc. 17th Int. Conf. Computers in Education (pp. 142–146).

Kodaganallur, V., Weitz, R., & Rosenthal, D. (2005). A comparison of model-tracing and constraint-based intelligent tutoring paradigms. *International Journal Artificial Intelligence in Education, 15*(2), 117–144.

Le, N-T. (2006) *A constraint-based assessment approach for free-form design of class diagrams using UML*. In K. Ashley, N. Pinkwart, C Lynch (Eds.) Proc. Workshop on Intelligent tutoring systems for ill-defined domains, 8th Int. Conf. ITS (pp. 11–19).

Le, N-T., Menzel, W. & Pinkwart, N. (2009) *Evaluation of a constraint-based homework assistance system for logic programming*. In S.C. Kong, H. Ogata, H. C. Arnseth, C. K. K. Chan, T. Hirashima, F. Klett, J. H. M. Lee, C. C. Liu, C. K. Looi, M. Milrad, A. Mitrovic, K. Nakabayashi, S. L. Wong, S. J. H. Yang (eds.) Proc. 17th Int. Conf. Computers in Education, APSCE (pp. 51–58).

Martin, B., & Mitrovic, A. (2002) *Automatic problem generation in constraint-based tutors*. In S. Cerri, G. Gouarderes, F. Paraguacu (Eds.) Proc. 6th Int. Conf on Intelligent Tutoring Systems (pp. 388–398).

Martin, B., Mitrovic, A., & Suraweera, P. (2008). ITS domain modelling with ontology. *Journal of Universal Computer Science, 14*(17), 2758–2776.

Mayo, M., & Mitrovic, A. (2001). Optimising ITS behaviour with bayesian networks and decision theory. *Artificial Intelligence in Education, 12*(2), 124–153.

Menzel, W. (2006). Constraint-based modeling and ambiguity. *Artificial Intelligence in Education, 16*(1), 29–63.

Mills, C. & Dalgarno, B. (2007) A conceptual model for game based intelligent tutoring systems. *ICT: Providing choices for learners and learning. Proc. ASCILITE* (pp. 692–701).

Mitrovic, A. (1997). *SQL-Tutor: a preliminary report.* Technical Report TR-COSC 08.97, Computer Science Department, University of Canterbury.

Mitrovic, A. (1998a) Learning SQL with a computerized tutor. *29th ACM SIGCSE Technical Symposium* (pp. 307–311).

Mitrovic, A. (1998b) *A Knowledge-based teaching system for SQL*. In T. Ottmann, I. Tomek (Eds.), Proc. ED-MEDIA'98, AACE, VA, 715–720 (pp. 1027–1032).

Mitrovic, A. (1998c) *Experiences in implementing constraint-based modeling in SQL-Tutor*. In B. Goettl, H. Halff, C. Redfield, V. Shute (Eds.), Proc. Intelligent Tutoring Systems (pp. 414–423).

Mitrovic, A. (2003). An Intelligent SQL tutor on the web. *Artificial Intelligence in Education, 13*(2–4), 173–197.

Mitrovic, A. (2005) *The effect of explaining on learning: A case study with a data normalization tutor*. In C-K. Looi, G. McCalla, B. Bredeweg, J. Breuker (Eds.) Proc. Artificial Intelligence in Education (pp. 499–506).

Mitrovic, A. (2012). Fifteen years of constraint-based tutors: what we have achieved and where we are going. *User Modeling and User-Adapted Interaction, 22*(1–2), 39–72.

Mitrovic, A. & Martin, B. (2000) *Evaluating the effectiveness of feedback in SQL-tutor*. In C. Kinshuk, T. Jesshope, Okamoto (Eds.) Proc. Workshop on Advanced Learning Technologies (pp. 143–144).

Mitrovic, A., & Martin, B. (2007). Evaluating the effect of open student models on self-assessment. *Artificial Intelligence in Education, 17*(2), 121–144.

Mitrovic, A., & Ohlsson, S. (1999). Evaluation of a constraint-based tutor for a database language. *International Journal Artificial Intelligence in Education, 10*(3–4), 238–256.

Mitrovic, A., & Ohlsson, S. (2006). Critique of Kodaganallur, Weitz and Rosenthal "A comparison of model-tracing and constraint-based intelligent tutoring paradigms". *Artificial Intelligence in Education, 16*(3), 277–289.

Mitrovic, A. & Suraweera, P. (2000) *Evaluating an animated pedagogical agent* In G. Gauthier, C. Frasson, K. VanLehn (Eds.), Proc. Intelligent Tutoring Systems, Springer (pp. 73–82).

Mitrovic, A., & Weerasinghe, A. (2009) *Revisiting the Ill-definedness and consequences for ITSs*. In V. Dimitrova, R. Mizoguchi, B. du Boulay, A Graesser (Eds.). Proc. Artificial Intelligence in Education (pp. 375–382).

Mitrovic, A., Djordjevic-Kajan, S., & Stoimenov, L. (1996). INSTRUCT: modeling students by asking questions. *User Modelling and User-Adapated Interaction, 6*(4), 273–302.

Mitrovic, A., Martin, B., & Mayo, M. (2002). Using evaluation to shape ITS design: results and experiences with SQL-tutor. *User Modeling and User-Adapted Interaction, 12*(2–3), 243–279.

Mitrovic, A., Martin, B., Suraweera, P., Zakharov, K., Milik, N., Holland, J., & McGuigan, N. (2009). ASPI RE: an authoring system and deployment environment for constraint-based tutors. *Artificial Intelligence in Education, 19*(2), 155–188.

Mitrovic, A., Ohlsson, S., & Barrow, D. (2013). The effect of positive feedback in a constraint-based intelligent tutoring system. *Computers & Education, 60*(1), 264–272.

Oh, Y., Gross, M.D., Ishizaki, S., & Do, Y-L. (2009) *Constraint-based design critic for flat-pack furniture design*. In S. C. Kong, H. Ogata, H. C. Arnseth, C. K. K. Chan, T. Hirashima, F. Klett, J. H. M. Lee, C. C. Liu, C. K. Looi, M. Milrad, A. Mitrovic, K. Nakabayashi, S. L. Wong, S. J. H. Yang, (Eds.) Proc. 17th Int. Conf. Computers in Education, APSCE (pp. 19–26).

Ohlsson, S. (1992) Constraint-based Student Modeling. *Artificial Intelligence in Education, 3*(4), 429–447.

Ohlsson, S. (2008) Computational models of skill acquisition. In R. Sun (Ed.), *The Cambridge handbook of computational psychology.* Cabmridge University Press. (pp. 359–395)

Ohlsson, S. (2015) Constraint-based modeling: from cognitive theory to computer tutoring – and back again. *Artificial Intelligence in Education* (this issue).

Ohlsson, S., & Langley, P. (1988). Psychological evaluation of path hypotheses in cognitive diagnosis. In H. Mandl & A. Lesgold (Eds.), *Learning issues for intelligent tutoring systems* (pp. 42–62). New York: Springer.

Petry, P.G. & Rosatelli, M. (2006), *AlgoLC: A learning companion system for teaching and learning algorithms*. In M. Ikeda, K. Ashley, T.-W. Chan (Eds.), Proc. ITS 2006, LNCS 4053 (775–777).

Poitras, G. J., & Poitras, E. G. (2013). *Computer-based learning software for engineering students*. Proc. Canadian Engineering Education Association.

Riccucci, S., Carbonaro, A., & Casadei, G. (2005) *An architecture for knowledge management in intelligent tutoring systems*.' Cognition and Exploratory Learning in Digital Age. Proc. IADIS Int. Cong. (pp. 473–476).

Roll, I., Aleven, V., & Koedinger, K. (2010) *The invention Lab: Using a hybrid of model tracing and constraint-based modeling to offer intelligent support in inquiry environments*. V. Aleven, J. Kay, J. Mostow (Eds.): ITS 2010, Part I, LNCS 6094 (pp. 115–124).

Rosatelli, M., & Self, J. (2004). A collaborative case study system for distance learning. *Artificial Intelligence in Education, 14*(1), 1–29.

Shareghi Najar, A., & Mitrovic, A. (2013). *Do novices and advanced students benefit differently from worked examples and ITS*?. In L. H. Wong, C.-C. Liu, T. Hirashima, P. Sumedi, M. Lukman (Eds.), Proc. Computers in Education (pp. 20–29).

Shareghi Najar, A., Mitrovic, A., & McLaren, B. (2014) *Adaptive support versus alternating worked examples and tutored problems: Which leads to better learning?* In V. Dimitrova et al. (Eds.) Proc. User Modelling, Adaptation and Personalization (pp. 171–182).

Siddappa, M. & Manjunath, A. S. (2008) *Intelligent tutor generator for intelligent tutoring systems*. Proc. World Congress on Engineering and Computer Science (pp. 578–583).

Suraweera, P., & Mitrovic, A. (2004). An intelligent tutoring system for entity relationship modelling. *Artificial Intelligence in Education, 14*(3–4), 375–417.

Wang, T., Mitrovic, A. (2002) *Using neural networks to predict student's behaviour*. In Kinshuk, R. Lewis, K. Akahori, R. Kemp, T. Okamoto, L. Henderson, C-H Lee (Eds.) Proc. Computers in Education (pp. 969–973).

Zakharov, K., Mitrovic, A., & Ohlsson, S. (2005) *Feedback Micro-engineering in EER-Tutor*. In C-K Looi, G. McCalla, B. Bredeweg, J. Breuker (Eds.) Proc. Artificial Intelligence in Education, IOS Press (pp. 718–725).

Zakharov, K., Mitrovic, A., & Johnston, L. (2008) *Towards emotionally-intelligent pedagogical agents*. In B. Woolf et al. (Eds.) Proc. Intelligent Tutoring Systems (pp. 19–28).

Zinn, C. (2014) *A lean constraint-based system to support intelligent tutoring*. Proc.14th IEEE Int. Conf. Advanced Learning Technologies (pp. 52–53).