Computer Intelligent Tutoring System "SQLTOR"

Ievgen Vagin[1], Olena Havrylenko[1], Juan Pablo Martínez Bastida[1], Andrey Chukhray[1]

[1] National Aerospace University, KhAI, Kharkiv, Ukraine
{ie.s.vagin, lm77191220, jpbastida, achukhray}@gmail.com

Abstract. Intelligent tutoring systems are required in different spheres, especially in IT. The presented system "SQLTOR" provides supporting tools for teachers and an adaptive tutoring approach for SQL students as well. Tutoring course in SQLTOR consists of task sequences ordered by complexity. Clustering and ordering are automatically performed based on student's degree of mastery of the relevant knowledge components in the learning domain. Thus, course structure allows gradually increase or decrease tasks complexity during tutoring process. SQLTOR provides hints which depend on learner's mistakes and the task content. Hints are automatically generated based on comparison of a student's SQL query with the referred one or are manually customized. The structure of SQLTOR, task grouping (clustering), ordering methods, SQLTOR tutoring modes and its behavior when a student makes mistakes are also described in this paper. As a conclusion, testing results from a group of students at National Aerospace University "Kharkiv Aviation Institute" are provided.

Keywords: Intelligent Tutoring System authoring tool, SQL computer learning, clustering, competence components.

1 Introduction

Implementation of computer tutoring programs (CTP) is one of the highest priority directions in educational tools evolution. This fact is reasoned by numerous advantages of CTP usage over the classical approach: adoption for a particular student, wide possibilities of virtual modeling of real objects and processes, decrease in time and work efforts for completion, verification of tutoring courses, e-learning facilities, etc.

Intelligent Tutoring Systems (ITS) are characterized by supporting inner and outer tutoring loops [1], minimal feedback (prompting hints and advices), nonlinear learning path, dynamic and customizable knowledge base, and self-learning support [1, 2]. ITS usually include three main structural elements: a domain model, a student model and a pedagogical model [2, 3], but researchers also often incorporate an interface model as the fourth element [3]. Special software is usually developed to overcome ITS development difficulties and it is commonly called as ITS Authoring Tools [4]. Users of ITS Authoring Tools need only basic knowledge in computing and minimal programming skills. Examples of ITS Authoring Tools are: DIAG, RIDES, SIMQUEST, XAIDA, Demonstr8, D3, TRAINER, ASPIRE, GTE, REDEEM, Eon, Interbook, MetaLinks,

CALAT, CTAT [3-5], etc. An actual problem in ITS development is a problem of clustering tutoring tasks. Clustering of tasks provides additional possibilities for the pedagogical model to select a next task according to the gradual difficulty incremental principle. A further problem is to provide adaptive hints in the context of certain subject area. A hint must be generated according to the place and type of student's mistake and assumes possibility of alpha and beta errors.

2 Research Purposes

ITS Authoring Tools must be an easy-for-using tool to simplify ITS development process. SQLTOR is a computer software designed with a "thick" architecture that allows building a distributed system by using existed data transfer protocols without requiring synchronizing system database between different computers.

Common structural elements of ITS are discussed in [3], they utilize specific elements of knowledge – Knowledge Components (KCs) [1, 5]. In the SQL tutoring domain, solutions are usually represented as queries. A query is registered as a solution of related tasks by applying one or more KCs. An example of the possible domain model structure is represented on Fig. 1.

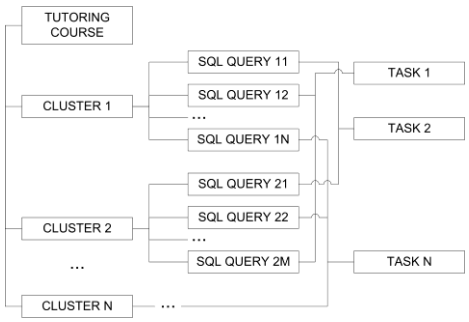

Fig. 1. Structure of the tutoring course in SQLTOR

Clusters on Fig. 1, are used to define groups of similar tasks. Similarity of tasks can be estimated from the intersection between KCs sets. KCs sets can be allocated by etalon queries registered as solutions for a particular task. Complexity can be decreased by means of automated clustering. General ITS development stages are shown on Fig. 2. Available author actions are represented by rectangles, arrows shows order of actions and rounded rectangles sign conditions for actions.

Student knowledge can be represented as a set of value pairs $\{KC_i, l_{KC_i}\}$, where KC_i – knowledge component and l_{KC_i} – mastering degree estimation of KC_i. Values of l_{KC_i} can be changed in accordance with the student success. Correct answers cause increment of the corresponding l_{KC_i} values and incorrect ones cause a decrement.

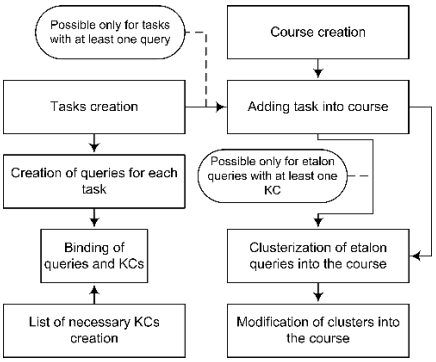

Fig. 2. Common development stages of ITS in SQLTOR authoring tool

3 Clustering of Tasks

All data required by SQLTOR is stored in two databases. The first of them is used for executing user queries and retrieve resulting rows to compare with the expected results. Second database is used to store data required for making tutoring decisions depicted on Fig. 3.

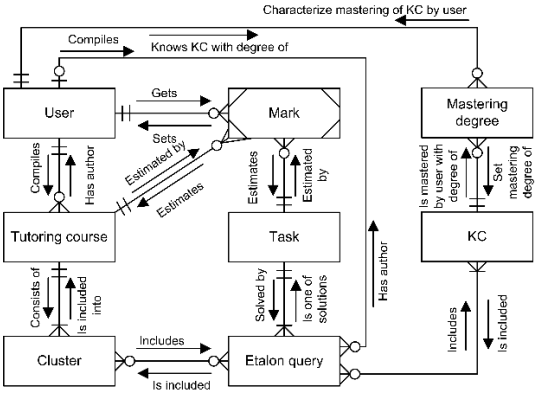

Fig. 3. ER-diagram of SQLTOR system database

Clustering algorithm in SQLTOR uses a specific method of f_D calculation that can be used for estimation of Q_i and C_j dissimilarity. Hamming distance can be used for such purpose: $h_{ij} = \sum x_{ic} \oplus x_{jc}$, where $\oplus$ is a binary XOR operation. Queries that have equal similarities with two or more centroids will be set into a cluster with less amount of etalons at the current algorithm iteration. Let specify distance calculation as follows:

$$D_i^j = f_D\left(Q_i, C_j\right) = h_{ij}\left(Q_i, C_j\right) + \left|\Theta_j\right| / \left|\Theta\right| = \sum_{c=1}^{n} x_{ic} \oplus y_{jc} + \left|\Theta_j\right| / \left|\Theta\right|, \qquad (1)$$

where h_{ij} is a function for calculating distance between the i^{th} query and j^{th} centroid; $|\Theta_j|$ is the number of queries in the j^{th} cluster; $|\Theta|$ is total number of queries. Correction $|\Theta_j|/|\{Q_i\}|$ is required to provide even distribution of queries in clusters.

Another peculiarity of SQLTOR clustering algorithm is related to the calculation of $f_C(\Theta_j)$. Each variable y_{jc} in tuple $C_j^* = f_C(\Theta_j)$, must be true, with a probability proportional to the number of true x_{ic} variables included in queries $Q_i \in \Theta_j$. We can obtain frequency of the c^{th} KC inclusion in queries of a j^{th} cluster as:

$$F_{jc} = \left(1/\left|\Theta_j\right|\right) \cdot \sum_{x_{ic} \in Q_i \in \Theta_j} x_{ic} \,. \tag{2}$$

Sum of x_{ic} is the number of true variables $x_{ic} \in Q_i \in \Theta_j$. Variable y_{jc} is initialized by "1", if frequency F_{jc} is greater than certain value of threshold frequency F'. Otherwise, y_{jc} must be initialized by "0" when $F_{jc} \le F'$. Thus, we can perform calculation of C_j^* as follows:

$$\begin{aligned} C_j^* &= f_C\left(\Theta_j\right) = \left\langle f_C^*\left(\{x_{i1}\}\right), ..., f_C^*\left(\{x_{in}\}\right)\right\rangle, \\ f_C^*\left(\{x_{ic}\}\right) &= \begin{cases} 1, & \text{if } F_{jc} > F', \\ 0, & \text{if } F_{jc} \le F', \end{cases} \quad x_{ic} \in Q_i \in \Theta_j. \end{aligned} \tag{3}$$

4 Task Ordering in the Outer Loop

Outer tutoring loop is responsible for selecting a next task in the educational flow. SQLTOR manages tasks in order to provide a more effective scenario for the passing course. SQLTOR can only use complexities for KCs, complexities of queries and tasks can be automatically calculated. Complexity of etalon query Q_i is calculated as follows:

$$P_{Q_i} = \sum_{c=1}^{n} P_{x_{ic}}, \quad x_{ic} \in Q_i, \tag{4}$$

where P_{Q_i} is complexity of query Q_i; $P_{x_{ic}}$ is complexity of c^{th} KC (if c^{th} KC is not in relation with query Q_i, then $P_{x_{ic}} = 0$). Complexity of tasks is defined as maximal complexity of queries that are associated with a task:

$$P_T = \max_{Q_i \in T} P_{Q_i}, \tag{5}$$

where P_T is complexity of the task T; P_{Q_i} is complexity of query Q_i, which is one of the solutions for task T. Average complexity of etalon queries is also calculated in the same cluster:

$$P_{C_j} = \left(1/N_j\right) \cdot \sum_i P_{Q_i}, \quad Q_i \in \Theta_j. \tag{6}$$

5 Tutoring Method Description

SQLTOR analyzes student's solutions by means of Abstract Syntax Trees (AST), which can be built based on a SQL query syntax structure. This approach gives possibility to avoid α and β-errors in student's solutions analysis and automatically generate hints. AST is a tree whose leaf nodes are operands, and other nodes are operators. In order to construct the AST for a particular SQL query, let us assume language keywords, such as SELECT, FROM, and other as operators and consider the construction of an AST of SQL language for the following query example:

```
SELECT name, salary FROM employees
WHERE salary>1000 ORDER BY salary DESC
```

The root of the AST is a synthetic node and the analysis of the query string is made from the beginning to the end, therefore, first keyword is SELECT. Column names in the result will be the child nodes, similarly to SELECT and the operator FROM. Besides the WHERE clause has a sub-operator ">", which will be the children of the node WHERE, and its operands that will respectively be children of the ">" node. ORDER BY clause is processed like other sections, except the keyword DESC. Node corresponding to DESC keyword will be the leaf one. AST of etalon query is provided on Fig. 4. We consider one of the possible situations when student's solution is:

```
SELECT name, salary FROM employees ORDER BY salary DESC
```

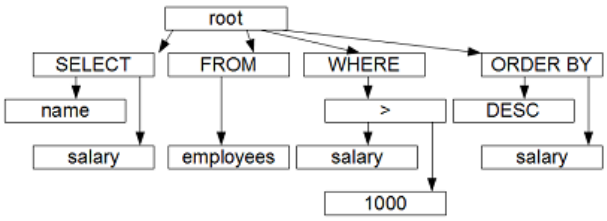

Fig. 4. Etalon query AST constructing

Nodes at current level h_i are compared pairwise in consideration of parent node at each iteration of the algorithm. In the case of inequality of nodes, student AST is considered as wrong. This search strategy is the most effective, because general query components are at high levels in the AST structure.

6 Conclusions

SQLTOR tests were performed on a group of 29 students. Tutoring course contained 47 tasks with 67 etalons. This list was extended by 25 new etalons after the testing process. New etalons were found from 9 students. All 29 students fully completed the course and solved the most complex tasks in each cluster. Testing outcomes exceed input testing results by 27% average. 16 of 25 students made mistakes in each task, but during output test, each student correctly solved at least two tasks. This indicates effectiveness of SQLTOR as a tool for supporting educational process on learning SQL programming language. Moreover, it provides the necessary creational tools for teachers of SQL courses. In addition, software provides ITS features to help students in acquiring SQL knowledge. It can be used for supporting learning process as well as self-learning and self-testing. System uses PostgreSQL database and is written in Java.

SQLTOR proposes to simplify work for authoring a tutoring course, it also includes several tools and features of tutoring tasks clustering, AST comparison algorithms of user and etalon solutions, represents knowledge as a set of knowledge components, supports sorting of tasks, etalons and clusters to provide a soft increment/decrement of the tutoring complexity.

References

1. VanLehn K.: The behavior of tutoring Systems. In: International Journal Of Artificial Intelligence In Education. vol.16(3), pp. 227-265. (2006)
2. Self, J.: The defining characteristics of intelligent tutoring systems research: ITSs care, precisely. In: International Journal of AI in Education. vol.10. pp. 350-364. (1999)
3. Beverly P. W.: Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning. In: Department of Computer Science, University of Massachusetts. pp. 480. Amherst , USA. (2008)
4. Murray T., Blessing S., Ainsworth S.: An overview of intelligent tutoring system authoring tools: Updated Analysis of the State of the Art. In: Authoring Tools for Advanced Technology Learning Environments. pp. 491-497. Kluwer Academic Publishers, Dordrecht, Netherlands. (2003)
5. Kulik A.S., Chukhray A.G., Pedan S.I., Ancenberger P.: Универсальная среда создания и трансляции интеллектуальных обучающих программ (Intellectual systems of decision-making and problems of computational intelligence). In: Proceedings of International Conference. vol. 1, pp. 189-192. Kherson, KNTU. (2009)
6. Mitrovic A.: Evaluation of a Constraint-Based Tutor for a Database Language. In: International Journal of AI in Education. vol. 10. pp. 238-256. (1999)
7. Intelligent Tutoring Systems, Chapter 37 / Corbett, Koedinger & Anderson / Chapter 37
8. Vygotsky L.S.: Психология развития человека (Phychology of human development). (eds.) “Смысл” publishing; ЭКСМО. pp.1136. (2005)
9. Parr T.: The Definitive ANTLR reference. Building Domain-Specific Languages. (eds.) Pragmatic Bookshelf. pp. 376. (2007)
10. Baxter I.D., Yahin A., Moura L., Sant’Anna M., Bier L.: Clone Detection Using Abstract Syntax Trees. In: Proceedings International Conference on Software Maintenance. vol. 8CB36272, pp. 368-377. (1998)