Scaffolding and fading problem selection in SQL-Tutor

Antonija MITROVIC and Brent MARTIN
Intelligent Computer Tutoring Group
Computer Science Department, University of Canterbury
Private Bag 4800, Christchurch, New Zealand
{tanja,brent}@cosc.canterbury.ac.nz

Abstract: Scaffolding is widely accepted as an effective strategy to use in learning environments. It is also agreed that for successful learning, scaffolding must fade to allow the learner to be in control of his/her own learning and acquire meta-cognitive strategies. Although scaffolding is used in almost all intelligent educational systems, fading is usually missing. In this paper we present an experiment whose goal is to study the appropriateness of fading. The study is performed in the context of the SQL-Tutor system, and the particular skill we focus on is the ability to select appropriate problems. We hypothesize that more able students would be better off when selecting problems on their own, and test whether this is valid. We also expect that for less able student, the most beneficial condition is faded problem selection: initially the system selects the problem for the student, giving explanations of why particular problems are good, and over time, the control over problem selection is given to the student. The results suggest that such an approach is effective.

1. Introduction

Adaptivity is central to many modern computer systems, especially Web-enabled ones. An adaptive system makes the user's task simpler or, in some cases, doable. Intelligent educational systems also adapt to each individual student's needs, learning abilities and preferences. However, there is an important distinction between educational systems and other applications whose goal is to support users in performing specific tasks. Intelligent educational systems must support the student in learning a task, and therefore should support all aspects of the task to be learned. In contrast, the goal of other types of adaptive applications is to support the user to perform a task faster or more efficiently. The support needed in educational systems therefore differs significantly from that needed by other kinds of adaptive systems. One of the crucial differences is that support in educational systems should fade over time, to allow the learner to resume control over the process, become independent and acquire metacognitive skills.

Intelligent educational systems provide support in the form of scaffolding, which is any kind of support the system provides to the learner in order to enable the learner to perform an activity, which is normally beyond their abilities. Scaffolding can be implemented in different ways: by providing problem-solving support, presenting instructional material at the right time and at the right level etc. Scaffolding in the form of adaptivity is central to Intelligent Tutoring Systems (ITS). It requires knowledge about the student to be collected and internally represented in the form of a student model. Scaffolding implies continuous assessment of the student in order to provide adequate support.

Although many projects showed (usually short-term) effects of scaffolding, recently researchers have started thinking about how much support is too much. Hubscher and Puntambekar [4] focus on adaptive hypermedia systems, and point out that adaptive navigation is a kind of scaffolding. The goal of any technique for adaptive navigation is to help learners find the relevant information. However, the authors claim that selecting a link is an important educational activity, and therefore students are robbed of the opportunity to learn this skill if navigation is always adapted. Other researchers investigate whether users prefer to be in control over how the system is adapted [5,6].

One skill important for learning is the ability to select a type of problem to practise on. This skill is related to self-assessment: in order to select the type of problems, the student should be able to identify gaps or misconceptions in his/her knowledge. In previous work, we have seen that less able students were generally worse at self-assessment than their peers, and also were less able to select appropriate problems to work on [7,8]. Studies show that having metacognitive skills results in improved problem solving and better learning [1,3], and that such skills can be taught [2]. We are interested in whether problem selection skills can also be taught.

Most ITSs select problems for each student based on the state of the student model (i.e. based on the student's knowledge), thus providing adaptive problem selection. In real life, it is very important for learners to be able to select an interesting problem to solve on their own. If ITSs always scaffold this activity, the student may not learn how to select problems. Therefore, in this project we wanted to explore scaffolded problem selection, and also decided to fade the scaffolding, in order to see what effects the different approaches would have on students' learning. To achieve that, we used three version of the same system: in the first version, problems are always selected by the system, in the second by the student, while in the third version the system initially helps select the problem for the student and explains the reasoning to the student, and over time the student starts selecting problem on his/her own. We also investigate the effects of these various problem selection options on the students of different abilities.

In the next section, we discuss the system we used in the study, and the three different versions of it, implementing various problem-selection strategies. In Section 4 we present our hypotheses and the design of the experiment, while the procedure used is described in Section 5. Section 6 presents the results, while the conclusions are given in Section 7.

2. SQL-Tutor and its versions used in the study

As discussed in the introduction, the purpose of this research is to investigate whether faded problem selection will have an effect on the student's problem selection skills. We performed an experiment in the context of SQL-Tutor, an intelligent tutoring system that teaches the SQL database language to university-level students. For a detailed discussion of the system, see [9]; here we present only some of its features. SQL-Tutor consists of an interface, a pedagogical module, which determines the timing and content of pedagogical actions, and a student modeller, which analyzes student answers. The system contains definitions of several databases, and a set of problems and the ideal solutions to them. SQL-Tutor contains no problem solver. To check the correctness of the student's solution, SQL-Tutor compares it to the correct solution, using domain knowledge represented in the form of more than 600 constraints. It uses Constraint-Based Modeling [10,11] to model knowledge of its students.

In SQL-Tutor, student may select problems in several ways: they may work their way through a series of problems for each database, ask the system to select a problem on the basis of his/her student model, select a problem from the list, or select a type of problem

they want to work on while the system selects a problem of that type on the basis of their student model.

For this study, we developed three versions of the system, differing from each other in the problem selection strategy. We wanted to student to reflect on their knowledge, in order to identify the type of problem they have difficulties with. To support reflection, we open the student model to the users. The constraint base of SQL-Tutor is large, and therefore it is not possible to show the student's progress directly in terms of constraints. In previous work [8] we have opened the student model by collapsing the student model into six parts, which correspond to the six clauses of an SQL query. Figure 1 presents a screenshot from SQL-Tutor, showing the page for problem selection. The student model is displayed to the student in terms of the progress over the six clauses. To measure progress on a clause, we compute the percentage of constraints relevant to the close that the student has used so far. The student model tracks how the student has used each constraint, and there is an estimate of the student's understanding of the constraint based on the last *n* uses of that constraint. We use these estimates to compute how well the student knows all the constraints relevant for the clause. The correctly known constraints are shown in green, while the ones the student has problems with are shown in red. The total shows the coverage of a particular constraint.

The three versions of the system used in the study support different problem selection strategies. In the first version, the system selects the appropriate type of problem for the

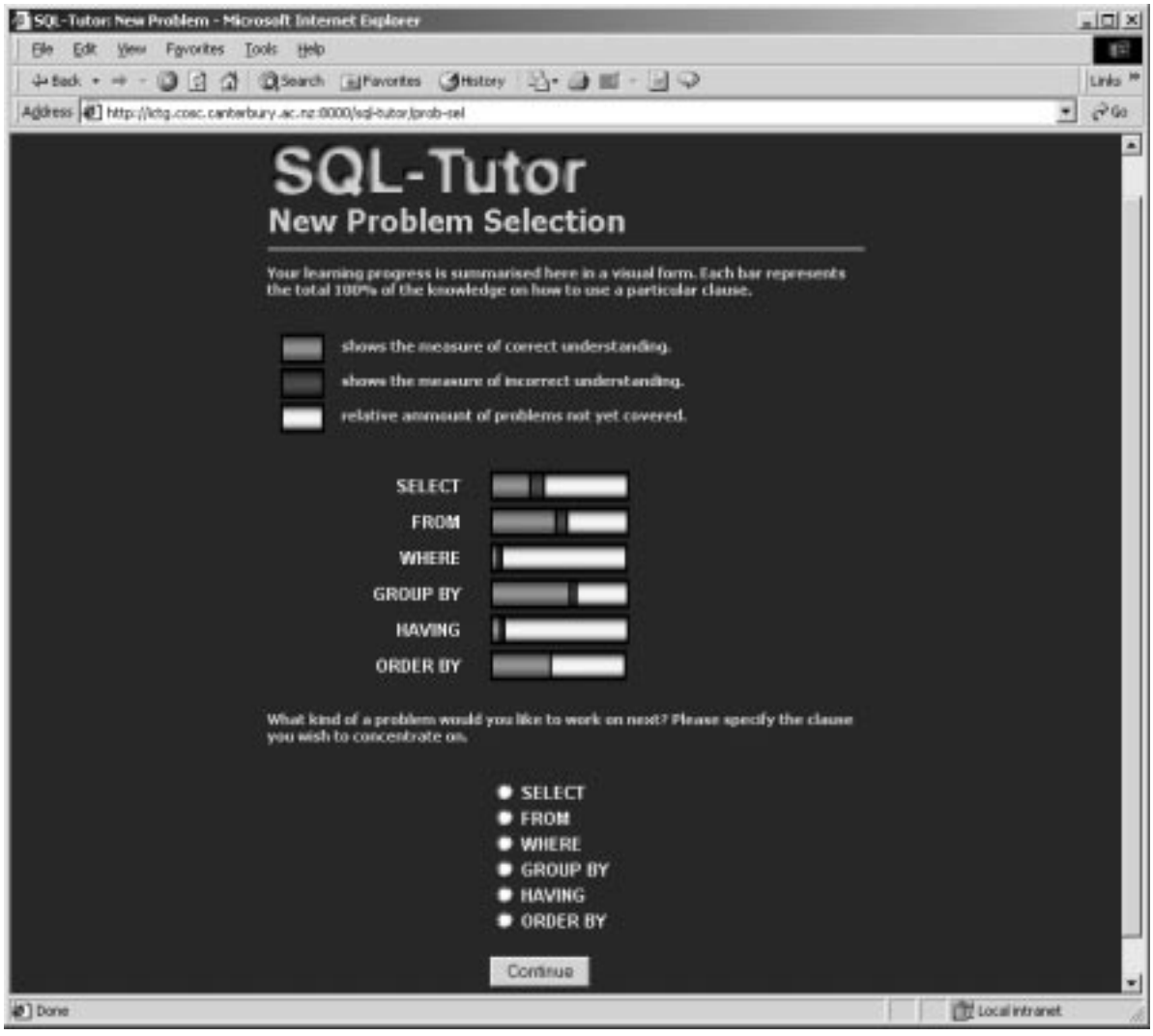

Fig. 1. The problem selection page from SQL-Tutor

student on the basis of the student model. When the student asks for a new problem, they get a page showing their student model, and a message specifying what type of problem is selected by the system.

In the second version, the student is always asked to select a type of problem, as in Figure 1. In the last version, the problem selection is faded. For novices, the student is asked to select the type of the problem, as in Figure 1. If the student's selection differs from what the system prefers, the student receives a new page, showing the student model and specifying the system's preference. Once the student's level increases over the threshold, the student is allowed to select the type of problems without system's intervention. Therefore we hypothesize that this version will support less able students in acquiring metacognitive skills, by opening the problem-selection strategy to them, and supporting reflection on their knowledge via the overview of their student models.

Once the type of problem has been determined in one of the previous three ways, the system searches for problems of the appropriate type that have not been solved yet. Out of the candidate problems, the system selects one that is at the appropriate level of complexity of the student.

3. Experiment Design

In previous work [7], we have seen that more able students are better at self-assessment than their less able peers. We hypothesized that more able students would also be better in problem selection, and therefore require less support. On the contrary, less able students would require more support in order to acquire such skills.

Table 1 summarizes the experimental design. We assess students' abilities by a pre-test. More able students are randomly allocated to versions where problems are selected by the student or by the system. Less able students are randomly allocated to one of the three versions of the system. We hypothesize that less able students would do the best in the faded condition, and do worst when selecting problems on their own. Also, we hypothesize that less able student will only be able to acquire problem-selection skills in the faded condition.

Ability	**Problem selection**		
More able	System	Student	N/A
Less able	System	Student	Faded

Table 1. The five groups

4. Procedure

We performed an evaluation study with the students enrolled in an introductory database course at the University of Canterbury, New Zealand, in the second half of 2002. Participation in the experiment was voluntary. Prior to the experiment, all students listened to three lectures on SQL and had one lab on the Oracle RDBMS. During the experiment, there were two additional lectures on SQL, and a series of four more labs. SQL-Tutor was demonstrated to students in a lecture on 16th September 2002. The experiment required the students to sit a pre-test, which was administered during the same lecture. The pre-test consisted of three multi-choice questions, each containing the text of a query, and some solutions. The students were asked to classify the solutions as correct or incorrect. The maximum mark for the pre-tests was 9. The students who sat the pre-test were given user accounts to use in SQL-Tutor from September 21st. Since the goal of the experiment was to investigate problem selection skills in combination with the students' abilities, students

were randomly allocated to one of the five groups, working with three different versions of the system, as shown in Table 1, after being classified as more or less able based on their performance in the pre-test.

The course involved a test on SQL a month after the system was introduced to the class, which provided additional motivation for students to practise with SQL-Tutor. The post-test was administered online the first time a student logged on to the system on or after 13th October 2002, and consisted of three questions of similar nature and complexity as the questions in the pre-test.

5. Results

Out of 167 students enrolled in the course, 76 sat the pre-test on September 16. Subsequently, some students who did not attend that lecture asked for usercodes and sat the pre-test. A total of 100 students completed the pre-test and were given access to the system. Table 2 gives the number of students in each group, their pre-test scores, and some additional information about their logs. The mean score for the pre-test for the whole class was 5.09 out of 9 (SD=1.69). The students were first divided into more able (42 students who scored 6 or more points on the pre-test) and less able (58 students), based on their performance in the pre-test. Then they were randomly allocated to one of the versions of the system, forming groups of similar size. A t-test showed no significant differences between the pre-test scores for the groups of same abilities, meaning the groups are comparable.

Participation was voluntary, and 35 students who sat the pre-test did not log on to the system at all. Table 3 gives the number of students in each of the groups that actually used the system (*Accounts used*). However, some of these students looked at the system only briefly. We excluded the logs of 4 students who did not attempted any problems, and the number of valid logs is given in the table. The remaining 61 logs were then analysed. The more able students solved more problems than the less able ones. The maximal number of solved problems was 160 for more able, and 65 for less able students. There is no significant difference between the numbers of problems solved for the groups of same abilities.

Group	Students	Pre-test mean (SD)	Accounts used	Valid logs	Solved problems
More able - system	21	6.81 (0.98)	16	14	30.65 (31.61)
More able - student	21	6.62 (0.80)	13	12	34.92 (42.65)
Less able - system	19	3.84 (0.96)	14	14	15.78 (17.89)
Less able - student	19	3.84 (1.21)	8	7	17.85 (14.19)
Less able - faded	20	4.05 (0.94)	14	14	14.5 (13.19)

Table 2. Some statistics about the groups

Table 3 gives the results on the pre- and post-test for students who have sat both. The two more able groups achieved higher results on the pre-test than on the post-test, but the difference is not significant. In previous studies with SQL-Tutor, more able students either improved [8] or achieved slightly lower scores on the post-test [7].

All three less able groups improved on the post-test, but the improvement is significant for the *faded* group only. This supports our hypothesis that less able students are not good in problem selection, and therefore would learn more when they do not need to select problems by themselves.

Group	Post-test	Pre-test mean (SD)	Post-test mean (SD)
More able - system	6	7.17 (1.17)	5.83 (1.47)
More able - student	6	6.67 (1.03)	5.17 (1.94)
Less able - system	6	3.33 (0.52)	4.67 (1.86)
Less able - student	3	3.67 (1.15)	4 (2)
Less able - faded	9	4.22 (0.97)	5.55 (1.51)

Table 3. Pre/post test results

5.1 Can students select their own problems?

We hypothesised that more able students would be able to select their own problems, and therefore the system should leave them to do that. If this is true, we might expect that for the more able group, students who selected the next clause themselves would perform as well as or better than those students where the system chose, while for the less able students those with system assistance would fare better. In fact this turned out to be false: those students with system assistance performed best, in terms of both their raw final score and the gain between the pre- and post-test.

We therefore measured the students' ability to select the clause, by comparing their selections with the system responses, and calculating for each group the mean number of times they chose the same clause as the system. Table 4 lists the results, with standard deviations in parentheses. Although the more able students appear slightly better at clause selection, the result is not significant. This indicates that in fact all students are fairly poor at problem selection.

Table 4. Clause selection ability

Group	**Clause selection accuracy (SD)**
Less able students	0.188 (0.195)
More able students	0.257 (0.195)
Significant? (2-tail T-test)	NO ($p = 0.43$)

The fifth group was coached in clause selection until they achieved a proficiency level for SQL of 4. To see whether or not this group learned the problem selection strategy, we measured how often they selected the same clause as the system for each of the first three levels. Figure 2 plots the results. This graph indicates that they did indeed learn the selection strategy, and that by the time they reached level 3 their proficiency at selection was much higher than the less able students who were not coached (65% vs 19%). However, the difference between levels 1 and 3 is not statistically significant ($p=0.135$)

Finally, we looked at the performance of the experimental group to see whether they continued to perform better at problem selection than their unaided counterparts. Unfortunately, we the problem selection strategy used by the system coincidentally *changes* at the same point that support is faded (prior to fading, only the first three SQL clauses are considered; afterwards all six are used), so the students have not been directly coached in the new selection strategy. We were therefore unable to determine whether or not the students continued to use their new problem selection skill after fading.

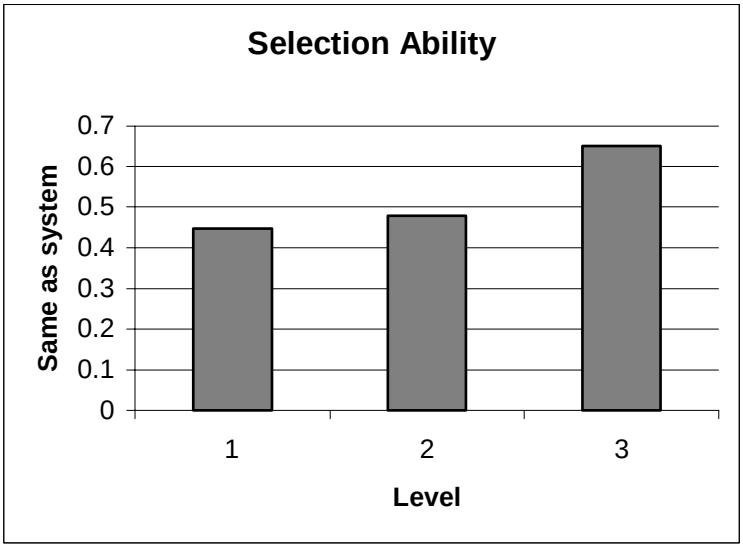

Fig. 2. Problem selection ability for coached students

6. Conclusions

The aim of this experiment was to assess students' problem selection abilities, and the application of faded coaching to this skill. We reasoned that since more able students appear better at self-assessment, they would probably also be better at problem selection. In contrast, less able students would be poorer at problem selection and would therefore perform best when the system selected the problems for them. We also submitted that students might be coached in this skill, which would have benefits when undertaking self-paced study away from the system.

We investigated problem selection skills in the context of SQL-Tutor, an intelligent tutoring system that teaches SQL. Three versions of the system were developed for this study. The problem selection strategy in two versions was fixed: in one version students always selected problems, while in the other one problems were selected by the system. The third version implemented faded problem selection. For less able students, the system initially selected problems while opening the problem selection strategy to the students. Over time, as the student's level increases, the system releases problem selection to the student.

The experimental results did not support our first hypothesis: more able students appeared to be no better at problem selection than their less able counterparts, with *all* students benefiting from system assistance at problem selection. However, the results *did* highlight the importance of problem selection: students that had system help performed best on the post-test. It also appears that attempts to coach students in the skill of problem selection were successful: the students in the faded group improved their selection accuracy, and performed better at selection than the students who were not coached.

All students were poor at problem selection despite having the student model available, suggesting that this is a worthwhile skill to teach. The faded approach taken here is probably appropriate, although the decision to fade should be based specifically on selection ability rather than general ability in the domain, since the results suggested the two were not strongly correlated.

Intelligent Tutoring Systems have suffered from the problem of limited skills transfer to the real world. One approach to overcoming this is to reduce the amount that students are scaffolded by the system so that they become more self-reliant, and fading is a means of achieving this. We have shown that problem selection is an important skill, and that it is

possible to teach students to select problems for themselves, allowing the system to fade its support. This is a step towards giving students more control over the teaching environment while at the same time coaching them in an important skill, and thus enriching their learning experience.

Acknowledgements

The work presented here was supported by the University of Canterbury grant U6430.

References

1. Aleven, V., Koedinger, K. (2000) Limitations of Student Control: Do Students Know When They Need Help? Proc. ITS'2000, Springer-Verlag, pp. 292-303.
2. Bielaczyc, K., Pirolli, P., Brown, A.L.: Training in Self-Explanation and Self-Regulation Strategies: Investigating the Effects of Knowledge Acquisition Activities on Problem-solving. *Cognition and Instruction*, 13(2) (1993) 221-252.
3. Conati, C., VanLehn, K.: Further Results from the Evaluation of an Intelligent Computer Tutor to Coach Self-Explanation. Proc. ITS'2000, Springer-Verlag, (2000) 304-313.
4. Hubscher, P., Puntambekar, S. Adaptive Navigation for Learner in Hypermedia is Scaffolded Navigation. In: P. De Bra, P. Brusilovsky, R. Conejo (eds) Proc. 2nd Int. Conf. On Adaptive Hypermedia and Adaptive Web-based Systems AH 2002, Springer, pp. 184-192, 2002.
5. Jameson, A. Schwarzkopf, E. Pros and Cons of Controllability: an Empirical Study. In: P. De Bra, P. Brusilovsky, R. Conejo (eds) Proc. 2nd Int. Conf. On Adaptive Hypermedia and Adaptive Web-based Systems AH 2002, Springer, pp. 193-202, 2002.
6. Kay, J. (2001) Learner Control. *User Modeling and User-Adapted Interaction*, 11, 111-127.
7. Mitrovic, A., Self-assessment: how good are students at it? In: J. Gilbert, R. Hubscher, S. Puntambekar (eds) *Proc. AIED 2001 Workshop on Assessment Methods in Web-Based Learning Environments & Adaptive Hypermedia*, San Antonio, pp. 2-8, 2001.
8. Mitrovic, A., Martin, B., Evaluating the effects of open student models on learning. In: P. de Bra, P. Brusilovsky and R. Conejo (eds) *Proc. 2nd Int. Conf on Adaptive Hypermedia and Adaptive Web-based Systems AH 2002*, Springer-Verlag LCNS 2347 (2002) pp. 296-305.
9. Mitrovic, A., Martin, B. and Mayo, M. Using Evaluation to Shape ITS Design: Results and Experiences with SQL-Tutor. Int. J. *User Modeling and User-Adapted Interaction*, v12no2-3, 2002: 243-279.
10. Mitrovic, A., Ohlsson, S.: Evaluation of a Constraint-based Tutor for a Database Language. *Int. J. on Artificial Intelligence in Education*, 10(3-4), (1999) 238-256.
11. Ohlsson, S.: Constraint-based student modeling. In: Greer, J.E., McCalla, G (eds): *Student modeling: the key to individualized knowledge-based instruction*, (1994) 167-189.