Evaluation of the Effectiveness of an Interactive Learning/Teaching Tool for Finite Automata

Xiao Yu Li and Matthias F. Stallmann
Computer Science Department
North Carolina State University

Microsoft Future Professors Project Report
Contents

1. Introduction

2. Project Overview and Implementation
   2.1 The ProofChecker Program
   2.2 Project Goals
   2.3 Procedures for the Comparison Study

3. Assessments
   3.1 Student Survey
   3.2 Performance Comparison
   3.3 Tutors’ Observations
   3.3 Teaching Assistants’ Feedback

4. Conclusions
   4.1 Future Plans
   4.2 Impact of Technology on Education
1. Introduction

As an important Computer Theory concept, Finite Automata has been taught as part of computer science core curriculum for over 30 years. Designing Finite Automata and their state-transition graphs contributes to the fundamental understanding of theoretical computer science and its wide range of applications. For a quick tutorial of the concept of Finite Automata and its design, please refer the power point slides “Coke Machine – An Simple Exam of Finite Automata”.

The process of designing the state-transition graph of Finite Automata involves drawing many circles representing the states and arrows representing the transitions. After seeing the soda machine example above, it is easy to imagine that the process of designing the state-transition graph can require many iterations, meaning that some circles and arrows may be erased and replaced before the final Finite Automata is produced. More importantly, the correctness of the Finite Automata can be hard to visually verified (by a grader, for example) because the complexity of the graph produced. To make matter worse, a Finite Automaton doesn’t have a unique state-transition graph representation so its correctness can’t be verified by simply comparing with a fixed key.

The design of Finite Automata has been traditionally taught with the pencil and paper approach. For the students, they often struggle with the concept because (1) the concept itself is highly abstract and mathematical, (2) there lacks a tool with which students can validate and guide their thought process during design. For the teachers, not only do they face the difficult material and the “disgruntled” students, they also often have a hard time grading student’s papers for the reason mentioned above.

We believe that with today’s technology and carefully written instructional software, we can significantly improve the way Finite Automata class is being taught and learned. In this project¹, we study the effectiveness of an interactive learning/teaching tool for Finite Automata called ProofChecker. In the fall of 2003, we worked with 103 students from two sections of CSC333, the undergraduate Automata course offered by NC State University. They are divided into two groups, one studied with ProofChecker and the one without. Our study is mainly based on comparison and contrast of the two groups in terms of student surveys, data analysis of the their performances, tutors’ observations as well as teaching assistant surveys.

In the next section, we give an overview of the project and how we conducted our study. We then present the outcome of the project and our assessments. Last, we conclude with some thoughts and future plans.

¹ This is a joint work of Suzanne Balik (principal investigator), Matthias F. Stallmann (faculty advisor and course instructor), Xiao Yu Li (Microsoft Fellow), and Susan High (designer of ProofChecker).
2. Project Overview and Implementation

The ProofChecker Program

In this project, we want to study whether and how we can enhance the learning and teaching of Finite Automata by introducing an internally developed interactive computer program - ProofChecker. ProofChecker attempts to address some of the problems of the paper and pencil approach by allowing a student to (1) design Finite Automata with an easy-to-use computer program (2) obtain instant validation and verification on the design process from within the program (3) save the work and submit it electronically. ProofChecker is also a grading tool, as its name implies. ProofChecker is written in JAVA; thus, it runs on all major computer platforms, meaning a student can use the program on home PC running Windows XP, school computer labs running Sun OS, laptops under Mac OS and also Tablets.

Project Goals

We set up three goals for this project:

(1) Study whether and how ProofChecker can enhance student’s learning performance in a traditionally pencil and paper oriented class.
(2) Study the impact of technologies on the learning experience of the students.
(3) Study the impact of technologies on the teaching experience, mainly in terms of grading.

Our first two goals will be assessed based on the performance comparisons of the two groups of students and the student survey. The third goal will be assessed based on the teaching assistant survey.

Procedures for the Comparisons Study

Since this study involves human objects (the students), it was subject to the school’s approval. We went through the necessary paperwork and obtained the permission to conduct such a study. Our comparison study consists of four major steps:

Step 1: Generate the control group and the experimental group

We worked with around 110 students who took the undergraduate Automata Theory course in the fall of 2003. First we asked the students to fill a survey on whether they are willing to participate in the study, their GPA and gender. 103 of them chose to participate in the study. The students were divided into two groups, one control group with 52 students (not using the ProofChecker software) and one experimental group with 51
students (using the software). The two groups have approximately the same distribution of GPA and gender.

**Step 2: Organize separate labs for each group**

We organized separate lab exercise sessions for each group. For the students in the control group, the labs were held in a regular lecture room, where students used the traditional paper and pencil approach to learn the class material on Finite Automata, with the help of one tutor.

For the students in the experimental group, the lab session was held in a computer lab, where each student was provided with a workstation and the ProofChecker program. A tutor was also available for assistance in the lab.

During a period of one week, eight 2-hour lab sessions (four for each group) were held with approximately 13 students in each session. The small size not only allowed us to fit the entire lab participants in the same computer lab or classroom, but also provide the opportunity for the tutor to interact with each student on a one-on-one basis. Both groups are given the exactly the same set of problems.

**Step 3: Homework assignment and exam grading**

Shortly after the lab sessions, the students were given regular homework problems similar to the exercises they did in the labs. The students in the experimental group were encouraged but not required to use ProofChecker for their homework. Significant portion of the midterm exam and final exam also tested students on the same material. However, since both the midterm and the final were paper-based, ProofChecker couldn’t be used during the exams. With the help of the TA’s for the class, we gathered grades for the homework and midterm problems pertaining to Finite Automata.

**Step 4: Student survey, data analysis, tutors’ observations and TAs’ feedback**

In this step, we first surveyed the students on their learning experiences and also got their feedback on the labs sessions they participated in. We then did statistical analysis to evaluate and compare the performances of the two groups based on their scores. We also record some interesting observations made by the tutors on the two types of labs. In the spring of 2004, we made ProofChecker the mandatory tool for doing and submitting some of homework. We invited the two teaching assistants, who also TAed for the same course in the fall of 2003, to share their experiences of the ProofChecker program as a grading tool.
3. Assessments

Our assessment of the project are based on four sets of data/feedback we have collected:
1. The student survey.
2. The performance comparison based on seven homework and test problems.
3. The observations made by the lab tutors.
4. The feedback by the teaching assistants.

Student Survey

In the student survey, we asked each student to answer three questions and we summarize the responses here:

(1) What aspects of the lab session did you find most useful (and why)?

For the control group, almost all students responded that they enjoyed working in groups and being able to discuss openly with the classmates. One student said: “It helped me understand the concepts better and it was more fun that way”. Some students also found the tutor very helpful in explaining sample problems and giving feedbacks.

For the experimental group, the students like the fact that ProofChecker can verify whether their solution is correct. The ability to get instantaneous feedback helps to make the problem easier to solve. Some students liked it because they can work in their own pace.

(2) Are there ways in which the experience could have been improved (be specific)?

For the control group, many students showed their desires for more lab sessions throughout the semester. Many also thought the lab could be more interactive and the tutor could have done a better job by going over all the answers at the end of the lab.

For the experimental group, the students gave many valuable suggestions regarding the features and functionality of ProofChecker. Some also felt that since they don’t work in groups, more individual help from the tutor is needed.

(3) How has the experience affected your performance in the course so far?

The problems that were worked on in the labs are similar to the ones in homework and the exams; therefore, students felt that the labs were a good practice. However, since the two-hour lab was a one-time thing and the material it covered is only a small fraction of the course material, most students felt that it had little impact on their overall performance in the class.
Performance Comparison

With the help of the TA’s, we collected scores for three homework problems, two midterm problems and two final exam problems, all of which cover the material on Finite Automata (the course teaches materials other than Finite Automata). We computed the average and standard deviation on each problem for each group. The score comparisons between the two groups showed very little difference for all but one homework problem.

When we looked that particular homework problem more closely, we realized that the problem is hard to solve with ProofChecker because of its lack of expressiveness: it is hard to use the existing vocabulary in ProofChecker to describe the states in the Finite Automata stated in that homework problem. As a result, the experimental group with ProofChecker did worse on that problem. The data we collected as well as the detailed statistical analysis is available in the accompanying excel file.

Tutors’ Observations

Besides the student survey and the quantitative analysis presented above, the two tutors (Suzanne and Xiao) also made some interesting observations on the two types of labs. One striking difference between the two labs is that the lab with ProofChecker is much more lively and interactive (between the students and the tutor) than its counterpart: we see more raised hands with their questions, more smiles, laughs and even sighs and frustrations in the computer lab session. This was unexpected because it is intuitive to think that in the environment of a computer lab, the students will tend to be more isolated.

We think the main reason for a livelier atmosphere in the computer lab is because ProofChecker is able to provide instant feedback on the correctness of the design. With a positive feedback, the students taste the success right way and are encouraged to do the next problem with more confidence. With a negative feedback, the students become curious on what they did wrong and are likely to ask the tutor for help. In the regular paper and pencil setting, such instant validation is not available unless a student can get constant attention from the tutor, but this is clearly impossible.

Teach Assistants’ Feedback

Two of the reasons why homework grading is difficult for the Automata course are:

(1) Automata drew on paper are often hard to read and student’s handwritings are hard to recognize.

(2) For a given problem, the solution is not unique. Therefore, when the student’s answer doesn’t match the key, it doesn’t necessarily mean the answer is wrong. However, it is hard to manually verify that answer.
ProofChecker has the potential of addressing both problems. The TAs can retrieve the answers (a source file generated by ProofChecker) submitted by the students and grade them using ProofChecker. So there is no messy handwriting to read; in addition, ProofChecker can verify all the answers with the click of a button, hence, increasing the reliability of the grading process.

In the spring of 2004, we taught the same course again and this time we made ProofChecker available to all students and they are required to use ProofChecker to do some of their homework problems and submit the solutions electronically to be graded.

The potential advantages of grading with ProofChecker are confirmed by the TAs. We have two TAs who have TAed the course for two consecutive semesters. Last semester, all homework was graded using the paper and pencil approach and this semester, we switched to grading with ProofChecker. Both TAs were asked to give feedback on the following two issues:

(1) Efficiency
Both TAs agree that there is a speedup in term of the time it takes to grading the homework. One TA responded: “Grading with ProofChecker was mush faster than grading on paper, we took 2-3hrs to grade homework this semester, which was really fast compared to our usual 5-6hr grading session for homework’s.”

(2) Accuracy
Both TAs feel that ProofChecker helps to increase the fairness of the grading process. They no longer have to deal with ambiguous explanations and messy drawings and the ProofChecker program can verify the students’ solutions. However, since ProofChecker only gives a yes/no answer (correct vs. incorrect), it is difficult to give partial credits.

Overall, the TAs consider ProofChecker as the preferred method for grading.

4. Conclusion

Future Plans

Over the course of this project, we have gathered many suggestions from the students on how to improve the features and functionalities of ProofChecker. Some of the improvements we are planning include:

(1) Expand the vocabulary for describing Finite Automata
(2) Allow trace and animation
(3) Allow minimization of a given Finite Automata

We plan to use ProofChecker in the Finite Automata courses we teach and share our studies with other computer science educators in professional conferences. Next, we go beyond ProofChecker and this project to share our views on the impact of technology on education.
Impact of Technology on Education

We see two ways how technology can be used to improve education. The first way is to use technology to make education “faster and easier”:

(1) Automate and speedup the mundane tasks (e.g., automated grading tools such as ProofChecker, elimination of slow hand writing by PowerPoint presentation)
(2) Make the education more accessible (e.g., course website and emailing list).

More and more educators are using technology this way to minimize the tediousness in their instruction. For the students, since much of the class materials are available in an electronic format, it allows easy access for the students. For the instructors, they benefit from being able to reuse class material they developed for subsequence courses they teach. At NC State, class website, message board, emailing list and grade book are now the standard tools for instructing a course. The advantages of such tools are clear.

But for the same reason why the best dishes doesn’t come from a fast food restaurant, an education doesn’t necessarily becomes better when technology makes it easier and faster to access. Technology can clearly be misused for education. I have taken courses in which I stopped taking notes because the class notes in PowerPoint slides were downloadable from the course website. When a class turns into a slide show with multiple frames per minutes, the students can become very passive in learning.

We believe the second way of using technology to improve education is more fundamental and much more challenging: that is to improve the overall quality of the education. With or without the using of technology, we believe a successful teaching/learning environment should be one that can constantly challenge and encourage the students to think and question. We have seen in the computer lab sessions of the ProofChecker study, a much more interactive learning environment was created when the technology helps the students to get instant feedback of their progress.

But there is no fixed recipe for how to create a successful environment for learning. The challenge for the educators is to find innovative ways to use technology to create such an environment. It would be very interesting to see the outcome of all the other fellowship projects and how technology has been used in their discipline to improve teaching and learning.