DEVELOPMENT AND USER TESTING OF NEW USER INTERFACES FOR MATHEMATICS AND PROGRAMMING TOOLS, FOCUSING ON THE COQ PROOF ASSISTANT

by

Benjamin Berman

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Computer Science
in the Graduate College of
The University of Iowa

September 2014

Thesis Supervisor: Associate Professor Juan Pablo Hourcade

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Benjamin Berman

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Computer Science at the September 2014 graduation.

Thesis Committee: ______________________________
Juan Pablo Hourcade, Thesis Supervisor

____
Member Two

____
Member Three

____
Member Four

____
Member Five

TABLE OF CONTENTS

CHAPTER

1 INTRODUCTION 1

2 COQ AND THE NEED FOR IMPROVED USER INTERFACES . . 6

3 COQEDIT, PROOF PREVIEWS, AND PROOF TRANSITIONS . . 7

3.1 Software Description 7
3.2 Experiment Design, Results, Analysis, and Conclusions 7

4 KEYBOARD-CARD MENUS + SYNTAX TREE HIGHLIGHTING, APPLIED TO FITCH-STYLE NATURAL DEDUCTION PROOFS . 8

4.1 Keyboard-Card Menus 8
4.1.1 Motivation 8
4.1.2 Software Description 8
4.1.3 Experiment Design, Results, Analysis, and Conclusions . 8
4.2 Syntax Tree Highlighting 8
4.2.1 Understanding Syntactic Structure 8
4.2.2 Structure Editing 8
4.3 Combined System Description 8

5 RELATED WORK 9

6 SUMMARY AND CONCLUSIONS 10

REFERENCES 11

CHAPTER 1

INTRODUCTION

The general principle behind this dissertation is that in order to accomplish difficult tasks, one generally needs to make these tasks easy–for moving boulders you might want some leverage. The significance of this principle was demonstrated to me when, a long time ago, my cello teacher pointed out that the way to play a difficult passage of music is not simply to grit one's teeth and keep practicing but to also figure out how playing those notes could, for instance, be made less physically awkward by changing the position of one's elbow. In general, when it comes to "virtuosic" tasks–tasks that require large amounts of skill–it is easy to ignore this "making things easy" principle and focus on putting more time and effort into practicing or studying, even though following the principle is often a requirement for success. The major goal of this dissertation is to apply the principle in the context of the virtuosic tasks that are involved in interactive theorem proving with the *Coq* proof assistant[**?**][1]: I intend to show ways to make the difficult task of using Coq easier by improving the user interface. As well as solving serious usability problems for an important and powerful tool for creating machine-checked proofs, many of the techniques I am developing and testing are widely applicable to other forms of coding.

The research involved in this dissertation is described primarily in chapter 3 and chapter 4. Chapter 3 describes "CoqEdit", a new theorem proving environment

[1]"Proof assistant" and "interactive theorem prover" are synonymous

for Coq, based on the jEdit text editor. CoqEdit mimics the main features of existing environments for Coq, but has the important property of being relatively easy to extend using Java. Chapter 3 continues with a description of prototypes of two potential extensions to CoqEdit. The chapter concludes with a description of a user study examining how these two extensions help, and potentially hinder, novice Coq users.

Chapter 4 describes a third prototype extension. Although this extension has not been tested with users, it presents a new, *general* scheme for manipulating text that I hope may be built upon and widely applied. The scheme involves the combination of two relatively novel ideas: “Keyboard-Card Menus” and “Syntax Tree Highlighting”. I describe both these two ideas and how they may be combined to help introductory logic students using a particular subset of Coq.

There are several points that I hope will become clear in this dissertation. First is how the research makes a positive contribution to society. While Coq is a powerful tool, and is already being used for important work, its power has come at the cost of complexity, which makes the tool difficult to learn and use. Finding and implementing better ways of dealing with this complexity through the user interface of the tool can allow more users to perform a greater number, and a greater variety, of tasks. At a more general level, the research contributes to a small but growing literature on user interfaces for proof assistants.[2] The work this literature represents

[2]The research draws from and contributes to two generally separate sub-disciplines of computer science, namely programming languages theory and human-computer interaction. I have Professors Juan Pablo Hourcade and Aaron Stump to thank for facilitating, and being

can be viewed as an extension of work on proof assistants, which in turn can be viewed as an extension of work on symbolic logic: symbolic logic aims to make working with statements easier, proof assistants aim to make working with symbolic logic easier, and user interfaces for proof assistants aim to make working with proof assistants easier. These all are part of the (positive, I hope we can assume) academic effort to improve argumentative clarity and factual certainty.

In addition, generalized somewhat differently, the research will contribute to our notions of how user interfaces can help people write code with a computer. The complexities of the tool in fact help make it suitable for such research, since a) they are partly the result of the variety of features of the tool and tasks for which the tool may be used (each of which provides an opportunity for design) and b) the difficulties caused by the complexity may make the effects of good user interface design more apparent. Furthermore, although Coq has properties that make it very appealing for developing programs (in particular, programs that are free of bugs), it also pushes at the boundaries of languages that programmers may consider practical for the time-constrained software development of the "real world". However, if, as in the proposed research, we design user interfaces that address the specific problems associated with using a language, perhaps making the user interface as integral to using the language as its syntax, these boundaries may shift outward. This means that not only are we improving the usability of languages in which people already are coding, we are also expanding the range of languages in which coding is actually possible.

---

involved with, unusual interdisciplinary work.

The second point that I hope will become clear in this proposal is that this research will be an intellectual contribution, i.e. that the project requires some hard original thinking. User interface development is sometimes "just" a matter of selecting some buttons and other widgets, laying them out in a window, and connecting them to code from the back end. While this sort of work can actually be somewhat challenging to do right (just one of the hurdles is that testing is difficult to automate), the project goes well beyond this by identifying specific problems, inventing novel solutions, and testing these solutions with human subjects.

The third and final point is that this work is actually doable. Some of it has already been accomplished and the results will be described below. The remaining work I also describe below, in enough detail, I hope, to make it seem reasonably straightforward.

In the remainder of this proposal I will first give a description of Coq, including its significance, an example of theorem proving using the tool, a description of current user interfaces, and some usability problems that I find particularly striking. I will continue with a description of a survey (including its results) on user interfaces for Coq that was sent to subscribers to the Coq-Club mailing list. Then, in the heart of this proposal, I will describe jEdit, CoqEdit, three experimental extensions to CoqEdit, an associated user study, and a timeline for the work. As an interlude within the discussion of the extensions, I include the contents of a paper I wrote with Professor Juan Pablo Hourcade, on the development and testing of "keyboard-card menus", the use of which is a critical ingredient in the third extension. I will conclude with an overview of (some of the) related work.

CHAPTER 2

COQ AND THE NEED FOR IMPROVED USER INTERFACES

CHAPTER 3

COQEDIT, PROOF PREVIEWS, AND PROOF TRANSITIONS

3.1 Software Description

3.2 Experiment Design, Results, Analysis, and Conclusions

CHAPTER 4

KEYBOARD-CARD MENUS + SYNTAX TREE HIGHLIGHTING, APPLIED TO FITCH-STYLE NATURAL DEDUCTION PROOFS

4.1 Keyboard-Card Menus

4.1.1 Motivation

4.1.2 Software Description

4.1.3 Experiment Design, Results, Analysis, and Conclusions

4.2 Syntax Tree Highlighting

4.2.1 Understanding Syntactic Structure

4.2.2 Structure Editing

4.3 Combined System Description

CHAPTER 5

RELATED WORK

CHAPTER 6

SUMMARY AND CONCLUSIONS

REFERENCES