Final Paper

Andrew Carter and Beryl Egerter

April 17, 2013

1 Previous Research

A number of research studies and papers have talked about code comprehension and debugging. The following papers cover both reviews of debugging and code evaluation and as such are highly connected to our work.

Mosemann and Wiedenbeck investigate code comprehension by novices with respect to navigation method (?, ?). They gave participants two different problems and asked a number of questions. The students were assigned a navigation method, and evaluated based on the navigation method. They concluded that novices prefer sequential, and control flow, even though they have not been trained in the latter. This pertains to our study because it is a qualitative review of navigation methods on a single problem. Furthermore they combined the results of the problems because there was no interaction between program and navigation method, however we suggest that with more diverse methods there may be some interaction, but they only posed two different problems.

Fitzgerald, et al. discuss debugging methods among novices (?, ?). They note an increased reliance on online resources for debugging, something that we did not provide. Their results show that bugs are relatively easy to fix, and that determining how programmers discover where a bug is located is more useful. Some student strategies exhibited that we could not possibly replicate in our study are using println statements and tracing using a debugger. They do confirm that students often used a tracing strategy on a variety of different debugging problems.

Rajlich and Wilde talk about concepts both as code formations and domain knowledge (?, ?). They argue that experts do not understand the entirety of the code they are debugging but simply map their domain knowledge concept on to code concepts to narrow down the code areas they must look at. This mapping of domain knowledge to code occurred in interview ACBE3 as the student connected functions within the code to actions taken in a connect four game. These mappings allowed the student to focus only on a single function instead of understanding the entire program.

Mayrhauser and Vans notice that often program comprehension involves multiple ways of thinking about the code (?, ?). In addition to explaining an umber of them and how they believe they work together, they perform a study and provide results of how often each of the strategies was used by a programmer looking for a section of code within a 40000 line program in a two hour period of time. While most of our interviews were far too short for our subjects to switch between different strategies, we did occasionally see them, and this paper provides information about what might occur if our problems and interviews had been longer.

These papers are relevant to our work because they cover various studies about code debugging, evaluation, and the strategies used in either case. Our work builds on these papers by comparing debugging and evaluation strategies directly instead of treating them as separate topics.