Exploring the Gender Effect on Cognitive Processes in Program Debugging based on Eye-movement Analysis

Ting-Yun Hou[1], Yu-Tzu Lin[1], Yu-Chih Lin[2], Chia-Hu Chang[3] and Miao-Hsuan Yen[4]
[1]*Graduate Institute of Information and Computer Education, National Taiwan Normal University, No.162, Sec. 1, Heping E. Rd., Da-an Dist., Taipei City 10610, Taiwan*
[2]*Department of Biomedical Engineering, Yuanpei University, No.306, Yuanpei St., HsinChu, Taiwan 30015, Taiwan*
[3]*Graduate Institute of Networking and Multimedia, National Taiwan University, Taipei City, Taiwan*
[4]*Graduate Institute of Science Education, National Taiwan Normal University, Taipei City, Taiwan*

Keywords: Computer Programming, Eye Tracking, Cognition Process, Program Debugging, Sequential Analysis.

Abstract: This study addresses the gender differences of cognitive processes involved in program debugging. In the experiment, twenty-five participants were asked to find bugs in the test programs. Eye-movement analysis was employed to track the students' gaze paths while they traced and tried to debug the programs. Cognitive processes were then obtained by employing sequential analysis of gaze data to investigate the significant sequences of attention areas. Cognitive processes of different genders were investigated by comparing the tracing sequences of program debugging. The experimental results show that both genders had limited working memory capacities for debugging the iterative program with complex computation. But females needed more manual calculation for the recursive program in this study. For the iterative structure, females tended to grasp the program requirements and then trace into the major part of the program, while males traced the change of output value according to the logic of the iterative statements. For the recursive problem, females traced the flow of recursive induction and the stop condition to execute the program and find bugs, while males traced the recursive function in a more leaping way. This study leaks the gender differences of cognitive processes in program debugging, based on which instructors/researchers can develop adaptive computer programming instruction for students of different genders.

1 INTRODUCTION

Many instructors/researchers investigated how to effectively and efficiently develop students programming skills. However, it is still challenging in the field of computer science education (Costelloe, 2004). The most challenging thing for novice programmers is to locate and resolve bugs (Lahtinen et al., 2005). Novices usually felt frustrated when they had to find bugs because their knowledge about programming was fragile (Perkins and Martin, 1986). Some research discussed programmers' debugging behaviors and found that novice programmers debug programs in a trial-and-error manner without comprehending programs so that they usually cannot resolve errors successfully (Fitzgerald, et al., 2008). Thus, investigating factors of students' cognitive processes in debugging programs is one of critical keys to improve their programming skills. However, cognition is a very complex process. Traditional research on cognition often conducted interviews and paper tests (Chen et al., 2010), but the results of these methods might be affected by missing memory, the exactitude of introspection, and unconscious social desirability. More than 80% of cognitive processes are obtained through vision (Sanders and McCormick, 1987). Eye movements are not smooth and individuals do not gaze on just one point of a visual field; they switch between fixation and saccadic eye movements in a very short space of time. Such complex brain activities underlie the concept of cognition. Through the process of eye-tracking, it is possible to obtain data regarding the amount and moment of eye fixation, saccade, gaze duration, regression, skipping, and refixation in an area of interest (AOI), which in turn can help us understand the cognitive processes occurring simultaneously. Eye movements detected by eye-trackers can provide much information regarding visual cognitive processes (Just and

Carpenter, 1984) and has been used to investigate creativity, learning, reading, teaching, affection, and problem solving. In a previous study, researchers applied eye-tracking to capture visual attention strategies and to conduct a detailed account of visual attention during a debugging task (Bednarik, 2011). Another study by Bednarik et. al. (Bednarik, Myller, Sutinen, and Tukiainen, 2006) reported that gaze can reflect the thinking mode of a subject, represent their interest level, and the importance of each visualized area. However, they did not find any correspondence between program debugging and the mental model used.

In this study, we investigated the gender effects on the cognitive processes of program debugging, which can give suggestions to instructors for design adaptive instructional strategies for different genders.

2 METHODOLOGY

2.1 Participants

The participants were twenty-five subjects (13 males, 12 females) of the Department of Computer Science in a university in North Taiwan. They studied programming in C for at least one year. None of the subjects had major psychological or psychiatric disorders that may affect the experimental results.

2.2 Procedure

The C language was utilized in this study. Participants were provided with 2 100-lines C programs (one was the iterative structure and the other was the recursive structure), each had three semantic or syntactic bugs. The time limit for each problem was 10 min. The subjects had to try to find all bugs by watching the source codes without using program development software. After they finished debugging the codes, they then had to explain the logic and the purpose of the programs in an interview in order to assess whether they understood the programs successfully had to try to find all bugs by watching the source codes without using program development software. After they finished debugging the codes, they then had to explain the logic and the purpose of the programs in an interview in order to assess whether they understood the programs successfully.

2.3 Data Collected

The Eyelink 1000 eye-tracker was employed to record participants' program debugging paths. The viewing distance was approximately 86 cm and the screen resolution was 1024 × 768. The gaze data was at first down sampled to obtain quantized gaze sequences. Sequential analysis (Bakeman, 1986) was then further employed for data interpretation and simplification. This algorithm can analyze how a set of complex probabilities behaved when it is applied repeatedly over time based on Maximum likelihood estimation. After sequential analysis, students' cognitive processes could be extracted.

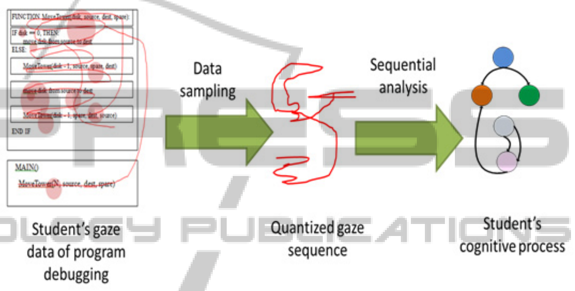
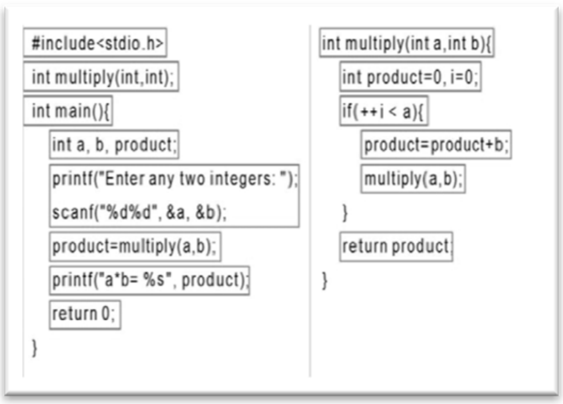

Data sampling
Sequential analysis
Student's gaze data of program debugging
Quantized gaze sequence
Student's cognitive process

Figure 1: System overview.

The whole experimental procedure, including the interview was recorded with a video camera to avoid loss of information. We also provided the subjects with the function of free drawing on the screen so that they could write down their thoughts or computational processes. The Region of Interest (ROI) was decided according to the logic of the program, e.g., variable definitions, I/O statements, conditional statements, function calls, computations in the function, and the note area. Figure 2 presents an example of a test program and the corresponding ROIs (each ROI was labeled with a block).



```
#include<stdio.h>
int multiply(int,int);
int main(){
    int a, b, product;
    printf("Enter any two integers: ");
    scanf("%d%d", &a, &b);
    product=multiply(a,b);
    printf("a*b= %s", product);
    return 0;
}

int multiply(int a,int b){
    int product=0, i=0;
    if(++i < a){
        product=product+b;
        multiply(a,b);
    }
    return product;
}
```

Figure 2: One example of ROIs.

3 RESULTS AND DISCUSSION

3.1 The Iterative Problem

The experiment results for the iterative program, as illustrated in Tables 1, show that when debugging the iterative program, both males and females had the significant sequence of

- Note area→Note area

This implies that both genders could not manipulate computations completely mentally while tracing the program and needed the note area to assist program tracing. The reason might be that their working memory capacities were not enough to trace the program because mental arithmetic is constrained by working memory (Adams and Hitch, 1997; Mackintosh and Bennett, 2003). Female students had the following additional sequences:

- Variable definition→I/O statement
- I/O statement→I/O statement
- I/O statement→Inner loop condition

That is, females paid much attention to the basic definitions and program requirements, which are the major step to comprehend the program (Chen, 2007). Males had the additional sequences:

- Computation in the conditional statement→ Variable definition→Conditional statements for the output value
- Output→Output

They spend more time tracing the value changes according to the logic of the iterative statements of the loop. Previous research (Marzieh, Dave, & Colin, 2007) argued that the most common mistakes made by novice programmers occur in the loop statements. Therefore, male participants in this experiment also considered the bugs might occur in the loop.

3.2 The Recursive Problem

In the experiment of the recursive program, females had the significant sequences (as presented in Table 2):

- Recursive function name→Variable definition in the recursive function → Conditional statements in the recursive function
- Conditional statements in the recursive function → Conditional statements in the recursive function
- Computation in the recursive function → Computation in the recursive function → Recursive call
- Note area→Note area

The results show that females often found bugs in recursive statements and followed the recursive logic, which compiles with Sabah's research (2012) indicating that students have to trace the flow of recursive induction and the stop condition to execute the program and find bugs. However, males tended to confirm the program requirements at first by seeing the I/O statements. Their significant sequences were:

- I/O statement→I/O statement
- Return value of the recursive function → Computation in the recursive function
- Recursive function name→Variable definition in the recursive function

Table 1: Results of the sequential analysis for the iterative program debugging.

z	Variable definition		I/O statement		Outer loop condition		Inner loop Condition		Computation in the conditional statement		Conditional statements for the output value		Output		Note area	
	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M
Variable definition	1.84	-0.59	2.54*	-0.5	-0.6	0.54	0.99	1.56	0.07	-1.35	0.11	2.88*	-0.68	-0.3	-0.84	0.17
I/O statement	0.52	0.28	2.57*	1.87	0.17	-1.79	3.13*	1.37	1.29	-0.24	-1.36	-0.65	0.02	-0.17	-2.73	-0.52
Outer loop condition	0.15	-1.17	-0.35	-1	0.6	1.51	1.96	1.66	1.03	0.64	-0.4	0.84	-0.11	-1.47	-0.39	-0.21
Inner loop Condition	-0.29	-0.64	1.33	0.7	1.96	-0.18	1.89	1.82	0.98	0.34	-0.73	-0.32	-0.18	0.24	-1.81	-1.22
Computation in the conditional statement	-0.52	2.36*	-1.2	-0.24	0.11	0.64	0.19	-0.3	0.13	-0.87	-0.9	-0.81	1.71	-0.43	-0.17	0.24
Conditional statements for the output value	-0.95	-0.81	0.12	-0.65	-0.4	-0.4	-0.73	-0.85	-0.25	-0.27	0.3	3.32	1.9	1.5	0.6	-0.49
Output	1.48	-0.3	0.52	-0.57	-1.24	-1.03	-1.14	-1.25	-1.39	1.08	1.11	1.5	0.83	2.5*	-0.96	-0.72
Note area	-1.37	-0.92	-1.99	0.49	-1.21	-0.21	-1.81	-0.29	-0.83	-0.23	2.35*	-1.28	0.24	-0.72	3.56*	2.03*

*$p < 0.05$.

Table 2: Results of the sequential analysis for the recursive program debugging.

Z`	I/O statement		Recursive function name		Variable definition in the recursive function		Conditional statements in the recursive function		Computation in the recursive function		Recursive call		Return value of the recursive		Note area	
	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M
I/O statement	-0.75	2.2*	-0.66	1.19	0	-1.39	0.85	-1.88	0.31	-0.27	-0.15	-1.26	0.66	0.46	-0.42	-0.59
Recursive function name	-0.66	-0.43	-0.59	-0.52	2.5*	2.72*	0.46	-0.94	-0.23	-0.42	-0.96	0.2	-0.64	-0.83	-1.1	-1.4
Variable definition in the recursive function	1	-0.66	0.24	-0.06	-0.59	-1	2.06*	1.03	0.68	1.03	-1.45	1.25	-0.97	-0.93	-1.05	-1.57
Conditional statements in the recursive function	-1.36	-0.27	0.46	-0.42	0.39	2.41*	2.85*	0.43	0.32	0.43	0.61	0.51	-1.31	1.12	-1.8	-1.17
Computation in the recursive function	-1.27	-1.88	-1.12	-0.42	1.86	0.56	-1.43	1.45	2.08*	0.77	2.58*	0.51	-0.4	1.12	-0.17	-1.17
Recursive call	1.71	-1.26	1.14	-1.32	-0.75	-0.8	-0.94	1.02	0.93	1.02	1.68	-0.6	-0.08	0.33	-1.78	-0.02
Return value of the recursive function	-0.72	0.46	-0.64	-0.83	-0.97	-0.93	-1.31	1.12	-1.22	2.71*	1.85	-0.85	-0.7	-0.54	2.18*	-0.9
Note area	`0.39	1.67	-1.1	-0.68	-1.05	-1.57	-1.35	-0.22	-1.62	-1.17	-0.65	-0.02	3.87*	0.21	2.91*	0.49

*$p < 0.05$.

・ Conditional statements in the recursive function →Variable definition in the recursive function

Males seemed to trace the recursive function in a leaping manner and did not pay equivalent attention to all recursive components. They traced the value changes according to the conditional statements and variable definitions in the recursive function. In addition, they did not use the note area as often as females did. It seems that males did not feel the limitation of their working memory for solving this recursive program.

4 CONCLUSIONS

This study addresses the gender effects on cognitive processes involved in computer program debugging based on eye-movement analysis. The naïve gaze data was analyzed by sequential analysis to find the sequences of cognition. The findings show that both genders had limited working memory capacities for debugging the iterative program with complex computation. But females needed more manual calculation for the recursive program in this study. For the iterative structure, females tended to grasp the program requirements and then trace into the major part of the program, while males traced the change of output value according to the logic of the iterative statements. For the recursive problem, females traced the program based on the recursive logic to find bugs, while males traced the recursive function in a more leaping way. The evidence of gender differences in cognitive processes of program debugging can help understand more about cognition of programming and design suitable programming instruction for different genders.

ACKNOWLEDGEMENTS

This work was partially supported by the National Science Council and the Ministry of Education of China under contact no. NSC 101-2511-S-003 -037.

REFERENCES

Adams J. W. and Hitch G. J., Working memory and children's mental addition. *Journal of experimental child psychology*, 1997;67(1): 21–38.

Bakeman R. *Observing interaction : an introduction to sequential analysis*. New York: Cambridge University Press, 1986.

Bednarik, R. (2011). Expertise-dependent visual attention strategies develop over time during debugging with multiple code representations. *International Journal of Human-Computer Studies, 70(2),* 143-155.

Bednarik, R., Myller, N., Sutinen, E., & Tukiainen, M. (2006). Program visualization: Comparing eye-tracking patterns with comprehension summaries and performance. *In ACM Proceedings of the 18th Workshop of the Psychology of Programming Interest Group*, 68-82.

Chen, H. C., Lai, H. D., & Chiu, F.C. (2010). Eye tracking technology for learning and education. *Research in Education Sciences, 4*, 39-68.

Costelloe, E. (2004). Teaching programming: The state of the art. *CRITE Technical Report, Centre for Research in I.T. in Education.*

Fitzgerald, S. (2008). Debugging: Finding, Fixing and Flailing, a Multi-Institutional Study of Novice

Debuggers. *Computer Science Education,* vol. 18, 93-116.

Just, M.A., Carpenter, P.A. (1984). Using eye fixations to study reading comprehension. *New Methods in Reading Comprehension Research. Hillsdale, NJ: Lawrence Erlbaum Associates*, 151-182.

Lahtinen E. (2005). A study of the difficulties of novice programmers. *SIGCSE Bull.,* 37, 14-18.

Mackintosh N. J. and .Bennett E. S, The fractionation of working memory maps on to different components of intelligence. *Intelligence*, 2003;31:519-531.

Marzieh, A. Dave, E., & Colin, H. (2005). An analysis of patterns of debugging among novice computer science students. *ITiCSE*, 84-88.

Perkins, D. & Martin, F. (1986). Fragile knowledge and neglected strategies in novice programmers. *Proceedings of the first workshop on empirical studies of programmers on Empirical studies of programmers,* 213-229.

Sabah AL-Fedaghi. (2012). Conceptual framework for recursion in computer programming. *Journal of Theoretical and Applied Information Technology, 46(2),* 983-990.

Sanders, M.S., & McCormick, E.J. (1987). *Human factors in engineering and design*. New York: McGraw-Hill.