Constructing a Model of Expert Parallel Programmers' Mental Representations Formed During Parallel Program Comprehension

Leah Bidlake **Eric Aubanel** **Daniel Voyer**
Faculty of Computer Science, Faculty of Computer Science, Department of Psychology
University of New Brunswick
leah.bidlake@unb.ca, aubanel@unb.ca, voyer@unb.ca

Abstract
Parallel programmers are frequently tasked with modifying, enhancing, and extending parallel applications. To perform these tasks and maintain correctness, parallel programmers must understand existing code by forming mental representations. The comprehension of parallel code requires programmers to mentally execute multiple timelines that are occurring in parallel at the machine level. The goal of the proposed research is to develop a model for parallel program comprehension. The study will investigate the mental representations formed by expert parallel programmers during the comprehension of parallel programs. The task used to stimulate the comprehension process will be verifying the correctness of parallel programs by determining the presence of data races. Eye tracking data and questionnaires will be used to formulate a model.

1. Introduction
During the comprehension process, programmers form mental representations of the code they are working with (Détienne, 2001). Understanding these representations is important for developing programming languages and tools that enhance and assist programmers in the comprehension process and other tasks. The cognitive component of program comprehension that is of interest here is the abstract mental representations that are formed during program comprehension. These mental representations, often referred to as mental models, are founded in the theories of text comprehension (Pennington, 1987). The mental model approach to program comprehension is based on the propositional or text-based model and the situation model that were first developed to describe text comprehension (Détienne, 2001).

2. Research Goals
In parallel programming there is a significant lack of theory to inform the development of programming languages, instructional practices, and tools (Mattson & Wrinn, 2008). Empirical research on mental representations formed by programmers during program comprehension has been predominately conducted using sequential code. Studies involving parallel programmers are most often concerned with productivity (Hochstein et al., 2005; Ebcioglu et al., 2006).

The comprehension of parallel code requires programmers to mentally execute multiple timelines that are occurring in parallel at the machine level. Therefore, parallel program comprehension may require additional dimensions to construct a mental representation. The goal of the research proposed here is to develop a model for parallel program comprehension that is based on the abstract mental representations formed by parallel programmers during program comprehension.

3. Research Ideas
Parallel programming has introduced new challenges including bugs that are hard to detect, making it difficult for programmers to verify correctness of code. One type of bug that occurs in parallel programming is data races. Data races occur when multiple threads of execution access the same memory location without controlling the order of the accesses and at least one of the memory accesses is a write (Liao et al., 2017). Depending on the order of the accesses some threads may read the memory location before the write and others may read the memory location after the write. Data races are difficult to detect and verify as they will not appear every time that the program is executed. To detect data races programmers must understand how a program executes in parallel on the machine and the memory model of the programming language.

In the proposed study, participants will be assigned the task of determining if a parallel program contains a data race. Participants will then be asked to identify the location of the data race if they believe that one exists and their level of confidence in their answer. The task of searching for a data race will be used to stimulate the comprehension process and as a result the programmer will form an abstract mental representation of the program. To determine if a program contains a data race programmers must mentally execute the program. This requires understanding how the program would execute on the machine and the possible interaction of executing multiple timelines of the program in parallel. To study the comprehension process an eye tracker will be used. Data from the eye tracker that may be able to assist in creating a model of program comprehension would be the order the code is read in, the sections of code that are revisited, and how often and how long the programmer spends reading sections of the code. This information will help to reconstruct their process for understanding and inform how they model the code. Additional information will be collected from participants in the form of questionnaires that will be developed to gain insight into their mental representations and understanding of the code.

Participants will be expert parallel programmers. The study of experts is important for informing the development of programming languages, instructional practices, and tools. To perform research on expert programmers it is necessary to be able to determine if participants are in fact experts. There has been a lack of agreement among researchers on how expertise should be measured and as a result there remains no standard for measuring programmer expertise. The distinction between expert and experienced also needs to be established. Experience is a measure of time spent working in a particular field or performing a task but does not necessarily translate into expertise, which is a measure of performance (Ericsson et al., 2006). Another research activity is to develop a tool for assessing programmer expertise.

4. Conclusion

Detecting the presence of data races in parallel code requires understanding the memory model of the programming language and how the program executes in parallel at the machine level. Using this task to stimulate the comprehension process will likely result in the formation of abstract mental representations with additional dimensions compared to those formed during the comprehension of sequential code. Through eye tracking and questionnaires we will develop a model for the mental representations formed by expert parallel programmers. To determine the level of expertise of participants, criteria will need to be developed that evaluates their programming skills.

5. References

Détienne, F. (2001). *Software design-cognitive aspect.* Springer Science & Business Media.

Ebcioglu, K., Sarkar, V., El-Ghazawi, T., Urbanic, J., & Center, P. S. (2006). An experiment in measuring the productivity of three parallel programming languages. In *Proceedings of the third workshop on productivity and performance in high-end computing* (pp. 30–36).

Ericsson, K., Charness, N. E., Feltovich, P. J., & Hoffman, R. R. (2006). *The cambridge handbook of expertise and expert performance*. Cambridge University Press. doi: 10.1017/CBO9780511816796

Hochstein, L., Carver, J., Shull, F., Asgari, S., Basili, V., Hollingsworth, J. K., & Zelkowitz, M. V. (2005, 11). Parallel programmer productivity: A case study of novice parallel programmers. In *Supercomputing, 2005. proceedings of the acm/ieee sc 2005 conference* (p. 35–35). doi: 10.1109/SC.2005.53

Liao, C., Lin, P.-H., Asplund, J., Schordan, M., & Karlin, I. (2017). Dataracebench: A benchmark suite for systematic evaluation of data race detection tools. In *Proceedings of the international conference for high performance computing, networking, storage and analysis* (p. 11:1–11:14). ACM. (event-place: Denver, Colorado) doi: 10.1145/3126908.3126958

Mattson, T., & Wrinn, M. (2008). Parallel programming: Can we please get it right this time? In *Proceedings of the 45th annual design automation conference* (pp. 7–11). New York, NY, USA: ACM. doi: 10.1145/1391469.1391474

Pennington, N. (1987). Empirical studies of programmers: Second workshop. In G. M. Olson, S. Sheppard, & E. Soloway (Eds.), (pp. 100–113). Norwood, NJ, USA: Ablex Publishing Corp.