Chair of Decentralized Systems Engineering
Department of Informatics

A Dependency-Preserving-Compiler for Linux Concurrency

General information	**Advisor**	Dr. Rodrigo Rocha (Univ of Edinburgh), Dr. Soham Chakraborty (TU Delft), Prof. Pramod Bhatotia (TUM)
	Email	rrocha@ed.ac.uk, s.s.chakraborty@tudelft.nl, pramod.bhatotia@in.tum.de
	Date	03.11.2021
Type	Master / Bachelor / Guided Research	
Description	In recent times weak memory concurrency has emerged as a dominant programming paradigm for modern hardware. Modern hardware performs various optimizations such as out-of-order execution that result in weak memory concurrency behaviors. To exploit the performance opportunities, programming languages and libraries also introduce concurrency primitives that enable efficient concurrent programming. Programming languages like C/C++ provide platform-independent abstractions for writing weak memory concurrent programs. The language specifications have defined the C/C++ memory model and state-of-the-art compilers (e.g. GCC, LLVM) follow the memory model. In parallel, Linux also introduces its own concurrency primitives and memory model to gain Performance improvement. The C/C++ and Linux memory models differ in subtle ways. Linux enforces memory ordering based on dependencies (data, address, control) while,	

	in contrast, dependencies do not play any major role in C/C++ concurrency. As a result, the compilers which follow the C/C++ model remove false dependencies. However, these compilers may introduce subtle concurrency bugs in linux if it does not preserve the dependency based ordering. This compiler-correctness gap is a serious concern for the linux community. The goal of this project is to bridge this gap by developing a dependency-preserving-compiler (DPC) that preserves the required orderings in linux in the correct and optimal manner. Finally, we study the performance impact of DPC on the linux kernel.
Keywords	Linux kernel, weak memory consistency, LLVM, concurrency
Goals	**Concrete outcomes** 1. Understanding linux memory model 2. Analysis of existing LLVM optimizations 3. Modify LLVM to develop a DPC compiler 4. Performance evaluation on linux kernel **Bonus points** Formalization of linux concurrency and formal reasoning of compiler correctness
Prerequisites	**Compulsory** ● C/C++ ● Knowledge of compilers ● Willing to collaborate and communicate with other members **Preferred** ● Knows LLVM internals ● Knows/interested in formal methods

References

1. Frightening Small Children and Disconcerting Grown-ups: Concurrency in the Linux Kernel. ASPLOS 2018. https://dl.acm.org/doi/10.1145/3173162.3177156

2. Towards Understandings the Costs of Avoiding Out-of-Thin-Air Results. OOPSLA 2018. https://dl.acm.org/citation.cfm?id=3276506

Application process

Please send an email to the advisor including the following:

- Email subject: "Thesis application (DSE)"
- CV
- A copy of your transcript(s)
- A **motivation statement**, please include samples of your work that you are proud of (e.g., major projects, open-source contributions, Github page, etc.) and/or writing samples (e.g., your technical blog, project reports, etc.)