1 Context

Fifteen years ago, CompCert [7] demonstrated that the decades-old dream [8] of a verified compiler was within grasp. Over the years, Leroy et al. indeed built an end-to-end optimizing C99 compiler coded and proved correct in the Coq proof assistant [12]. The proof of correction establishes that the assembly program resulting from the compilation only exhibits dynamic behaviors already present in the source program: the compiler preserves functional correctness.

This seminal work spawned in its wake a vast body of works on verified compilation. Some have sought to develop similar techniques for other languages [4], others to extend the applicability of the result — w.r.t. linking in particular [9] — or yet others to extend the scope of the result — seeking to prove the preservation of more properties than mere functional correctness [1].

Vellvm is an ambitious project in this design space. It aims in the long run to formalize and verify part of the LLVM compilation infrastructure, and focus in particular its early efforts on the formalization of LLVM IR, the intermediate language at the core of LLVM. Beyond the specific language of interest, the modern iteration of the Vellvm project [14] has the originality to explore the use of modern semantic techniques based on the Interaction Trees [13], a recently introduced generic Coq library to design denotational domains. With these modern semantic tools come novel proof techniques in the realm of verified compilers, putting an emphasis on algebraic approaches.

Despite an extremely active decade, the verified compilation community has vastly danced around a major theme: concurrency. An interesting piece of work, CompCertTSO [10] has been developed in 2013: it extends CompCert’s sequential semantics with a synchronization machine to support a TSO-based concurrent executions. Although promising, this work remains a solitary anomaly in the domain. Recently, Boulmé et al. [11] verified a compiler with instruction scheduling, albeit in a restricted case where the parallelism involved did not impair determinism. This situation is particularly unfortunate in that concurrency is well known to be both a major source of bugs, as well as extremely difficult to debug. This feature is therefore ruled out in many critical environments. Much like CompCert made the use of optimizing compilers in critical contexts more amenable, a verified compilation chain supporting concurrency could pave the way toward the adoption of some forms of concurrency in similar contexts.

The lack of contributions in the realm of verified concurrent compilation is even more blatant when contrasted to modern formal works on the verification of concurrent programs. Rich bodies of works have been developed to formalize the complex memory models supported by languages such as LLVM IR [6]. Modern semantic methods have been introduced to model realist forms of concurrency [2]. Incredible advances in separation logic has made the mechanical verification of complex concurrent programs possible [3].

As a realist, modern development, Vellvm is the ideal soil to change this situation and develop verified techniques for the compilation of non-sequential programs. This PhD aims to initiate this ambitious line of work.

2 Objective of the PhD

This PhD thesis takes place in the context of a long term, ambitious research goal: developing at scale techniques and tools for the verified compilation of concurrent languages. We anchor in particular this objective around LLVM
IR and the Vellvm project. The objective of this PhD thesis is geared toward laying the necessary foundations for this endeavor, and exploring some of its ramifications.

Vellvm in its current state, owing to its semantic denotational domain built on top of the Interaction Trees library, enjoys strong properties of compositionality and modularity. Maintaining these properties when extending the semantics to multi-threaded executions is an open problem. The first natural objective of the PhD is therefore to propose an extension of Vellvm that does not necessarily entail such strong properties over concurrent executions, but does not jeopardize the existing results when applied to sequential programs.

These semantics concerns are complex. Interleaving sequentially consistent executions while preserving rich reasoning principles is already difficult, but LLVM IR relies on an extremely subtle weak memory model. While this model has been well studied during the last few years, there remain many open questions that must be resolved to extend a project such as Vellvm to support it. In particular: can axiomatic approaches be reconciled with the strong constraints on executability that we have? Are formal models currently proposed suitable for reasoning on the formal correctness of optimization passes? Given the complexity of this question, we stage the objectives of the PhD incrementally, by aiming first at building a sequentially consistent model on top of Vellvm, and refining it further in a second time.

Once Vellvm is equipped with a concurrent semantics, numerous directions of research open up.

Formal semantics are not only good for formally verifying tools and transformations. They can notably be at the basis of symbolic interpreters providing means to rigorously and finely test intermediate representations. Such tools are useful in general, but particularly so for concurrent programs where testing and debugging via compilation is extremely hard due to the inherent non-determinism. We would like to explore this theme as a more applied pendant of this project.

The proof of correctness of optimizations naturally requires new proof techniques when tackling a concurrent programming language. Furthermore, new optimizations become of interest, such as the minimization of the number of synchronizations performed by a program. These avenues are vastly left to be explored, which we propose to do.

Third, LLVM IR, as a low level intermediate representation designed for compilation, provides a very general paradigm for concurrency. High level languages however often offer paradigms that are less expressive, but easier to use in a safe way: future and promises are two examples of such popular constructions. Understanding how to correctly compile such paradigms down to Vellvm, and how to reason about the correctness of these compilation passes, is another major avenue that a concurrent support for Vellvm would open.

Without envisioning a full compilation chain yet, these first steps already constitute an extremely ambitious research program for a PhD. However, each of these steps constitutes a reasonable small project that can be addressed more or less thoroughly; the success of the PhD is thus not conditioned by the complete achievements of all these steps. We build on top of it a possible tentative program destined to guide, but not constrain, the PhD student through their work.

3 Program

The PhD may follow the following steps (some of the steps are independent and could be given relative priority based on subsequent findings and the candidate’s preference):

- Study of LLVM’s support for concurrency: synchronization primitives, memory model, specific optimizations;
- Naive extension of Vellvm’s semantics to support a simple Sequentially Consistent memory model;
- Design of a symbolic interpreter based on this semantics suitable for testing;
- Embedding and/or compilation of high level concurrent programming paradigms in Vellvm;
- Implementation of concurrent-centric optimizations;
- Formal proof of correction of these optimizations;
- Refinement of the semantics in order to support the actual weak memory model supported by LLVM IR.

3.1 Location and Supervising

The PhD will take place in LIP, Lyon, France. It will be supervised by:

- Yannick Zakowski, Inria researcher, main supervisor (70 % on this PhD). No current PhD advising or supervising. Yannick Zakowski is a specialist in verified compilation and the use of the Coq proof assistant; he is the lead developer of the current version of Vellvm and one of the designers of the Interaction Trees library.
• Matthieu Moy, assistant professor at Lyon 1 (30 % on this PhD). Currently co-advising two second year PhD students with Ludovic Henrio. Matthieu Moy is notably an expert in compilation in general, and in particular targeting multi-core architectures. He has also extensive experience with LLVM, notably in the context of abstract interpretation for LLVM IR and dedicated compilation techniques for SystemC.

This PhD is expected to be conducted additionally in tight collaboration with Ludovic Henrio, CNRS researcher in the CASH team. Ludovic Henrio is an expert in the semantics of many concurrent paradigms. He has extensive experience in the development of tools and transformations formally proved correct w.r.t. these semantics.

4 Sources of Inspiration

We list hereafter a few sources of inspiration that can support the development of a verified compilation infrastructure supporting concurrency for LLVM IR. They are by no means exhaustive.

• The description of the sequential fragment supported by Vellvm [14] is the natural basis for this line of work.
• CompCertTSO, an extension of CompCert to TSO memory models. In particular, their approach to building the concurrent semantics on top of the pre-existing sequential one via a synchronization machine could be a source of inspiration for a lighter first approach to the extension of Vellvm.
• The memory model supported by LLVM IR is subtle. The promising semantics [6] provides a modern formal account of it that should be a major source of inspiration once we decide to tackle it in its full complexity.
• Cerberus [5], developed at Cambridge, has proposed a formal semantics for the weak memory model supported by C11. Although this work is not embedded in a proof assistant, and hence not suitable for verification, it already provides an invaluable tool to both understand the semantics, as well as interactively step through the behaviors of a program for testing purpose. LLVM IR’s memory model being relatively close, although not identical, to C11’s memory model, this work should be a precious source of inspiration.

References


