# Abstract

In modern multiprocessors, the consistency of shared memory would be relaxed to increase the computing power; hence, the value of a memory location could be observed as different values at the same time on each execution unit. Note that, term memory model is usually used to determine the semantics of the memory system. In particular, the memory model that relaxes the consistency of the shared memory is usually called relaxed memory model. Consequently, an anomalous result of the concurrent programs could occur on relaxed memory models. Therefore, relaxed memory model is the primary concern to ensure the program correctness.

For ensuring program correctness, the program property is defined as the invariant of the concurrent programs. Due to the relaxed memory models, this research provides an abstraction, called operation structures, of the concurrent programs. The targets of this abstraction are (1) to be sufficient for program verification, and (2) can describe the essence of assembly programs to be verified. Consequently, the program verification approach should be introduced to prove the program property on target relaxed memory model. In particular, this research uses SMT-based program verification approach to ensure the program correctness automatically.

This thesis shows two program verification methods for relaxed memory models. Mainly, the methods rely on the SMT-based program verification approach. In both methods, the behavior of program execution and the program property are encoded into a verification condition represented by a first-order formula; the formula is then used to check every execution satisfies the program property. The primary difference between the proposed methods is the way to abstract the behavior of program executions into the verification condition.

In both methods, the program executions are abstracted symbolically. In particular, the computation of program execution is considered in SMT-based program verification. The first method uses the bounded loop unwinding technique to abstract the symbolic executions. In the bounded method, the loop iterations are unwound systematically within a bound. For the second method, the inductive invariant approach is used instead of loop unwinding. However, the proposed inductive invariant method has seemed to be sound for partial store ordering (PSO) and stronger memory models. For SMT-based program verification, the abstraction of program execution and the program property are encoded regarding the relaxed memory model into a first-order formula. Primarily, the encoded formula is a decidable formula to be solved by an SMT solver automatically. Consequently, the program correctness can be ensured automatically.

In the experiment, an experiment tool was developed, and the Z3 solver is adopted to solve the first-order formula. As a result, the tool can automatically verify the property of the abstraction of concurrent programs on a relaxed memory model. In particular, the abstraction of concurrent programs can represent some essential behaviors of assembly programs. Besides, the bounded method is an under-approximation approach, while the inductive invariant method is an over-approximation approach.

In summary, concurrent assembly programs can be abstracted for ensuring the correctness by our methods. For the bounded method, the program correctness on a relaxed memory model

can be ensured if there is no loop. Otherwise, the method can at least disprove the program property on a relaxed memory model. As for inductive invariant method, the correctness of concurrent program contains loop can be ensured on partial store ordering (PSO).

**Keywords**: Concurrent Program Verification, SMT-based Program Verification, Multiprocessors, Relaxed Memory Model, and Automated Program Verification.