An investigation of the proof score approach to formal verification and a new case study with the approach

1910005 Naoki Asae

Today, software systems permeate every part of our lives. Software systems have become an indispensable part of our lives. However, the number of reports of system failures caused by software systems has been increasing more and more. While the technology to develop software systems is advancing day by day, the technology to assure the quality of the software systems is not keeping pace. Since software systems are expected to play an increasingly important role in the future, there is an urgent need to establish new quality assurance techniques. One of the techniques to assure the quality of software systems is formal method. Formal methods can be divided into formal specification and formal verification, and formal verification can be further divided into model checking and theorem proving. Formal methods are one of the most promising techniques to assure the quality of software systems, but they are not yet widely used in software development. To make it popular, we need to increase the number of cases where formal methods are applied. Mutual exclusion is an important issue in the creation of secure software systems. The mechanism to achieve mutual exclusion is called mutual exclusion protocols. In this study, we formally verify that three mutual exclusion protocols (TAS protocol, Qlock protocol, and Anderson protocol) enjoy the mutual exclusion property. TAS protocol uses an atomic instruction test & set, Qlock protocol is an abstract version of the Dijkstra binary semaphore, and Anderson protocol is an array-based mutual exclusion protocol. For each protocol, the formal verification is conducted in two ways: (1) by writing proof scores in CafeOBJ, an algebraic specification language, and (2) by using CafeInMaude Proof Generator (CiMPG) and CafeInMaude Proof Assistant (CiMPA). Proof scores are programs written in CafeOBJ to conduct formal proofs. CafeInMaude is the world's second implementation of CafeOBJ in Maude that is a sister language of CafeOBJ. CafeInMaude is equipped with CiMPG and CiMPA. CiMPG takes proof scores and generates proof scripts that can be fed into CiMPA. While conducting the formal verification of Anderson by writing proof scores in CafeOBJ, we encountered a situation such that our proof attempt did not seem to be convergent: it seemed necessary to us an infinite number of similar lemmas. To tackle the situation, we have introduced an auxiliary variable into Anderson, where an auxiliary variable does not affect the behaviors of Anderson. We describe the situation in detail in the report. We have learned

some lessons from the case studies and summarize the lessons in the report. In particular, by tackling the formal verification of mutual exclusion protocols with two different approaches, manual proof by proof scoring and automatic proof by using CafeInMaude, I was able to understand the advantages and disadvantages of each approach. And I was able to understand firsthand why they have not yet penetrated into the field of software system development, although formal methods have been attracting attention as an effective technique for quality assurance of software systems. Since the purpose of this case study was to find out the reason, this is the most important lesson I learned from this case study. We also mention some pieces of our future work. For example, we do not know whether it is mandatory to introduce an auxiliary variable into Anderson so that we can formally verify that Anderson enjoys the mutual exclusion property. If so, we would like to clarify why we need to do so. Otherwise, we would like to complete the formal verification of Anderson without introducing any auxiliary variables.

Keywords: formal methods, theorem proving, mutual exclusion protocols, auxiliary variable.