The application of logic in theorem proving and system verification

Chu Duc Thang
Department of Computing Science
University of Alberta
thang@ualberta.ca

1 Introduction

An automated reasoning system is a computer system with the ability to search for proof of logical assertion[1]. The system becomes useful to verify the accuracy of the mathematical proof. In particular, it can even replace peer review with a more objective criterion. Furthermore, automated reasoning systems can extend the ideas of math verification to computer verification, so that the system can complement the standard testing procedure. There are various systems for solving the two issues above such as HOL[2], Isabelle[3], and Coq[4]. However, each reasoning system has unique functionality and is suitable for a given task. Therefore, this work will illustrate the usage of different automated reasoning systems as well as their strengths and weaknesses.

2 Types of automated reasoning systems

In this literature, various automated reasoning systems will be divided into two groups: human-like versus computer-oriented systems, and automated versus interactive systems.

First, a few researchers attempted to design a system modeling the way humans reason. For example, Allen Newell and Herbert A. Simon developed an automated method known as Logic Theorist [5] to establish several important theorems. Around that time, Gelerntner also designed a system to prove theorems in Euclidean geometry [6]. Both of these systems resemble humans' thoughts since they use heuristic functions, which are primarily the creation of human designers, to figure out which search path is most likely to result in the desired outcome. Besides the human-like approach, machine-oriented reasoning is another approach for automated system development. Robinson developed a prover based on a resolution method [7], whereas Davis and Putnam used Herbrand's theorem to design an enumeration process [8] to demonstrate if the first-order logic theorem is true. Maslov's inverse method [9] is another, less popular strategy. The primary distinction between human-like and machine-oriented approaches is that the latter use formal logic theories to establish the proof, which slightly deviates from human logic. Since human-like approaches often use heuristic functions, which can be completely wrong, the performance of machine-oriented system has an advantage over human-like systems in some domains.

Fully automated reasoning systems were first considered by researchers and achieved notable results. The EQP theorem prover[10], developed by William McCune to address the Robbin conjecture, is among the most well-known examples. Additionally, satisfiability modulo theories (SMT)[11][12], a fully automated system developed recently, have been combined with deduction in specific theories. However, some mathematical problems are so complex that fully automated reasoning systems are not able to solve them. Consequently, developing a proof-verification system where a human may offer assistance to the computer is a more sensible strategy. More specifically, the system can check the proofs against probable human error or derive some tedious parts of the proofs automatically. AUTOMATH[13], Mizar[14], HOL[2], and Isabelle[3] provers are a few prominent examples that have a significant impact in multiple aspects. Because of machine-oriented approaches' strengths over

human-like ones[15], this work will focus on the applications of machine-oriented automated systems and machine-oriented interactive systems in mathematical formalization and formal verification.

3 First application: Formalization of Mathematics

Mathematical formalization is the process of expressing a theorem statement in a formal language and using a set of formal inference rules to derive the theorem. The formalization of mathematics aims to eliminate human error potential. Since mathematics publications are reviewed by peers before publication, there are many instances in which published results, such as the four-color theorem[16], turn out to be incorrect. Therefore, three widely-known problems as well as the provers that solve them will be presented below to demonstrate the importance of the system.

EQP[10], which solves the Robbins conjecture, is one of the first well-known provers. In 1993, EA single binary and one unary operation made up the Robbins algebra that Edward Huntington introduced. This equation led to the long-standing but unproven conjecture that Robbins algebra is a subset of Boolean algebra. It wasn't until 1995 that William McCune used the EQP program [10] to prove this conjecture by arriving at the key lemma for the proof. EQP was based on first-order equational logic that performs well on lattice-like structures. EQP is very similar to another method called Otter[17] since both of them are based on first-order logic. Despite having a similar logical approach, the Otter system offers a more refined and stable solution than EQP.

HOL Light[2] and Isabelle [3] prover were combined to solve another well-known problem, the Kepler conjecture. Thomas Hales made the first attempt to prove the conjecture exhaustively using proof assistants. However, the panel of judges eventually rejected his work since the execution of a computer program is unverifiable. To improve the previous work and persuade the panel, Hales developed the Flyspeck project, which utilized a hybrid of HOL Light and Isabelle in its formal proof system[18]. Despite following the logic of computable function (LCF)[19] style, the HOL Light and Isabelle systems differ in a few significant respects. In particular, the HOL Light is an interactive system that provides a substantial library of already demonstrated mathematics. Meanwhile, Isabelle/Isar is an automatic method that focuses on broad logic applications and prioritizes human-readable formal proof.

Lastly, Coq[4] prover was developed to solve the Four-Color Theorem, which consequently indicates the drawbacks of peer-reviewed systems and the requirement for automated reasoning tools. After the discovery of mistakes made in Alfred's proof, Wolfgang Haken and Kenneth Appel presented the first method for using a computer-assisted prover[20][16]. The mathematical community, however, continued to suspect the evidence because computer-assisted proof cannot be manually verified by a human. Finally, Georges Gonthier proved the theorem using Coq [4]to address the previous doubt. The Coq prover eliminates the need to rely on particular computer programs to verify a specific situation by using the Coq kernel, hence resolving the prior uncertainty. Although Isabelle[3], HOL[2], and Coq are LCF-like provers, Coq follows Curry-Horward isomorphism and uses the Calculus of Inductive Constructions to prove. Coq's expressiveness enables users to define rich mathematical objects and manipulate the proof objects explicitly.

Besides the examples above, there are many mathematical theorems proved by logical systems. In particular, Mizar[14] has been extensively used to formalize several key theorems, including Fermat's Little Theorem, the Banach fixed point theorem, and the Hahn-Banach theorem. Mizar has several advantages over different methods since it allows natural language input with a large library of proven theorems. Although all of the theorem proofs do not have all of the mathematics theorems in their library, they all show how important it is to have automated proving methods.

4 Second application: Formal verification

Formal verification, in particular, is the process of mathematically confirming that computer systems behave correctly as expected. Although most programs start out with clearly defined logical concepts, it is exceedingly difficult to develop a computer program that always operates as intended, which leads to errors. To detect these bugs, extensive testing has been commonly used. However, this approach has the drawback that, in some cases, there are too many combinations of possibilities. The disadvantage of this strategy is that there may be too many possible combinations in some circumstances. If all possible combinations are not tested, a subtle error could go unnoticed. As

a result, rigorously proving the computer system gains appeal since it is exhaustive and enhances understanding of the system. Various automated and interactive techniques, including HOL[2], Isabelle/Issar[3], and Coq[4], are used as a solution to verify hardware and software errors.

A hardware bug refers to a defect in the design or operation of computer hardware and floating-point incorrectness is one of the frequent defects. For instance, an uncaught floating-point exception caused the Ariane 5 rocket to self-destruct[21] during its first launch and cost $500 million to rebuild. Another well-known case was the Intel Pentium P5 FDIV problem[22]. The floating-point division algorithm's missing value in the lookup table caused the floating-point incorrectness. Intel had to recall the faulty CPUs in late 1994, which cost them $475 million. Following that, Intel developed a comprehensive formal verification system for floating-point problems that included an interactive prover, HOL Light[2]. Meanwhile, AMD used the ACL2[23] prover to construct its own verification mechanism. ACL2, a highly automated system that solves first-order number theory problems without the need for explicit quantifers, is different from HOL. Strong engineering and comprehensive documentation are main reasons for the system's popularity. Moreover, ACL2 is based on an actual programming language that facilitates theory development, program execution, and debugging. In addition to systems that address floating-point problems, there are other systems that deal with various computer components. CompCert[24] uses Coq as the underlying technique to verify a compiler from a large chunk of the C programming language into a PowerPC assembler. In the meantime, HOL and Isabelle/Issar[3] are essential for confirming the L4 operating system microkernel version[25].

Logical verification techniques are used in software in addition to hardware components. NASA's SPIDER[26](Scalable Processor-Independent Design for Enhanced Reliability) uses the Prototype Verification System (PVS)[27] to validate the protocols. The system is built based on type theory without proof objects and is suitable for high-order logic theorems. Furthermore, PVS has been applied in file system verification[28] and real time systems[29]. B-method is another common approach due to its reliability. It is also used in safety-critical European systems like the automated Paris Metro Line 14[30].

5 Conclusion

It has been shown that automated reasoning systems are beneficial in a number of contexts, such as formal computer verification and mathematical formalization. This work provides a summary of popular systems and their applications in different domains. All of the systems shared some weaknesses in that their libraries do not prove all of the theorems. Also, since different reasoning systems work separately, sharing knowledge between systems can hide the weaknesses of an individual system. This work aims to inspire researchers to find out new applications in different domains or improve the current existing techniques to achieve a better result.

References

[1] F. van Harmelen, V. Lifschitz, and B. Porter. *Handbook of Knowledge Representation*. ISSN. Elsevier Science, 2008.

[2] M. J. C. Gordon and T. F. Melham. *Introduction to HOL: a theorem proving environment for higher order logic*. Cambridge University Press, USA, 1993.

[3] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[4] Y. Bertot, G. Huet, P. Castéran, and C. Paulin-Mohring. *Interactive Theorem Proving and Program Development: Coq'Art: The Calculus of Inductive Constructions*. Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg, 2004.

[5] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Program. Lang. Syst.*, 1(2):245–257, oct 1979.

[6] H. Gelernter. *Realization of a geometry-theorem proving machine*, page 134–152. MIT Press, Cambridge, MA, USA, 1995.

[7] J. A. Robinson. Theorem-proving on the computer. *J. ACM*, 10(2):163–174, apr 1963.

[8] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, jul 1960.

[9] S. Yu. Maslov. An inverse method for establishing deducibility of nonprenex formulas of the predicate calculus. 1967.

[10] William McCune. Solution of the robbins problem. *Journal of Automated Reasoning*, 19:263–276, 1997.

[11] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Program. Lang. Syst.*, 1(2):245–257, oct 1979.

[12] Robert E. Shostak. Deciding combinations of theories. *J. ACM*, 31(1):1–12, jan 1984.

[13] de Ng Dick Bruijn. The mathematical language automath, its usage, and some of its extensions. *Studies in logic and the foundations of mathematics*, 133:73–100, 1970.

[14] Andrzej Trybulec and Howard Blair. Computer assisted reasoning with mizar. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'85, page 26–28, San Francisco, CA, USA, 1985. Morgan Kaufmann Publishers Inc.

[15] H. Wang. Toward mechanical mathematics. *IBM Journal of Research and Development*, 4(1):2–22, 1960.

[16] Percy J. Heawood. Map-colour theorem. *Proceedings of The London Mathematical Society*, pages 161–175, 1949.

[17] William McCune and Larry Wos. Otter - the cade-13 competition incarnations. *Journal of Automated Reasoning*, 18:211–220, 1997.

[18] T.C. Hales. *Dense Sphere Packings: A Blueprint for Formal Proofs*. London Mathematical Society Lecture Note Series. Cambridge University Press, 2012.

[19] M. Gordon, R. Milner, and C.P. Wadsworth. *Edinburgh LCF: A Mechanized Logic of Computation*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1979.

[20] Alfred Bray Kempe. On the geographical problem of the four colours. *American Journal of Mathematics*, 2:193, 1879.

[21] Jacques-Louis Lions. Ariane 5 flight 501 failure: Report by the enquiry board. 1996.

[22] Thomas Kropf. Introduction to formal hardware verification. In *Springer Berlin Heidelberg*, 1999.

[23] Matt Kaufmann and J. Strother Moore. An industrial strength theorem prover for a logic based on common lisp. *IEEE Trans. Software Eng.*, 23:203–213, 1997.

[24] Xavier Leroy. Formal verification of a realistic compiler. *Commun. ACM*, 52:107–115, 2009.

[25] Gerwin Klein, Kevin Elphinstone, Gernot Heiser, June Andronick, David A. Cock, Philip Derrin, Dhammika Elkaduwe, Kai Engelhardt, Rafal Kolanski, Michael Norrish, Thomas Sewell, Harvey Tuch, and Simon Winwood. sel4: formal verification of an os kernel. In *Symposium on Operating Systems Principles*, 2009.

[26] Laure Vieu. Spatial representation and reasoning in artificial intelligence. 1997.

[27] Sam Owre, John M. Rushby, and Natarajan Shankar. Pvs: A prototype verification system. In *CADE*, 1992.

[28] Wim H. Hesselink and Muhammad Ikram Lali. Formalizing a hierarchical file system. *Formal Aspects of Computing*, 24:27–44, 2009.

[29] Natarajan Shankar. Verification of real-time systems using pvs. In *International Conference on Computer Aided Verification*, 1993.

[30] Susan L. Gerhart, Dan Craigen, and Ted Ralston. Case study: Paris metro signaling system. *IEEE Software*, 11:32–28, 1994.