定理証明とモデル検査

Theorem Proving and Model Cheking

藤田 憲悦
Ken-etsu Fujita

群馬大学大学院理工学府
Department of Computer Science, Gunma University

1 まえがき

定理証明システムとモデル検査技法は検証に不可欠な要素技術・ツール (reliability techniques and tools) となってきている．論理の二つの側面，演繹 (deduction) と意味論 (semantics) から定理証明とモデル検査の研究の流れを概観する．そして，定理証明とモデル検査技法の組合せ，確率的モデル検査・シミュレーション等の可能性について議論する．

2 定理証明

2.1 演繹的手法

Hoare は 1969 年の論文 [1] で手続き型プログラムに対して公理と推論規則からなる演繹的システム (deductive reasoning) を与えた．これは今後における

(i) プログラム検証 (Proofs of program correctness)

(ii) 言語の形式的定義 (Formal language definitions)

の基礎をなしている．前者は，公理と推論規則からプログラムの正当性を検証する演繹的推論を可能にしている (Hoare 論理)．

後者は，プログラミング言語の意味に対して形式的仕様を与えていおり (公理的意味論)，**実装者**と (熟練した) **ユーザー**の間の共通理解の橋渡しをしている．また，形式的意味論では言語のある部分を“ 未定義 ”のままにしておく事もできる (例えば，オーバーフロー，0 による除算の扱いなど)．これにより，ハイレベルな言語を様々な設計のハードウェアに効率的に実装できることが見込まれる．特に，Hoare が述べている項目 **(ii)** の形式的意味の定義は，公理的意味に限らず，最近の検証事例である CompCert(C コンパイラの形式的検証 [2]) と seL4(OS カーネルの形式的検証 [3]) においても重要な意味を持っている．そこでは，“ ユーザ ”**(ソースコード/抽象的仕様・モデル)** と“ 実装者 ”**(アセンブリ/カーネル)** の間で意味の受け渡しが整合的に行われている **(模倣関係にある/段階的に詳細化されている)** ことが定理証明器 [4, 5] を活用して演繹的に証明されている．

2.2 Propositions-as-Types

定理証明器といっても自動証明系から支援系まで様々なシステムがあるが，Coq[6] に代表される支援系は高階の型理論に基づいており，そこでは Propositions-as-Types(“ 命題 ”イコール“ 型 ”) の原理が興味深い．言い換えると，Proofs-as-Programs(“ 証明 ”イコール“ プログラム ”)，Normalization of Proofs as Evaluation of Programms (“ 証明の正規化“ イコール“ プログラムの評価“) を意味している．この原理は，論理と計算の世界をつなぐ一つの観点をあたえている．例えば，証明と論理式が整合的に対応しているかどうかを検査する問題は，プログラムの型チェック問題そのものである．特に高階型理論の場合では，型検査問題の決定可能性はプログラムに含まれる情報に依存している [7, 8].

Propositions-as-types 自体興味深いが，それ以上に，様々な論理 (古典論理，様相論理，線形論理) と様々な計算体系 (継続，分散計算，通信プロトコル) にもこの観点が汎用されていることは拡張の指針となっている [9].

定理証明のツールに関しては，Wideijk[10] は 17 の定理証明システムの調査・比較検討を行っている．また，JML と定理証明器を活用した Java の検証ツールの調査は永宮・藤田 [11] にもまとめられている．

3 モデル検査

3.1 モデル論的手法

モデル検査の概念とアイディアは 1981 年の Clarke-Emerson の論文 [12], 及び独立に Queille-Sifakis[13] で与えられた．それ以前で検証といえば，2.1 にある様に Floyd-Hoare[1] の枠組みに基づき逐次的なプログラムを演繹的手法で一つ一つ検証することが主流であった．

Pnueli[14] は，並行システムの検証のために線形時間時相論理 (LTL) を用いた演繹的手法を導入した．1981 年 1 月の POPL でも，BenAri-Manna-Pnueli[15] は分岐時間時相論理 (CTL) に対するヒルベルト流の演繹的体系を与えた．さらに有限状態機械での決定問題も扱っていたが，単項 2 階論理 (S1S) への還元に留まりモデル検査のアルゴリズムまでは至らなかった．しかし，**時相論理 (Temporal logic)** を活用してリアクティブシステムの性質を表現したことはモデル検査に繋がる大きな原動力となった．

その後間もなく 1981 年 5 月に，Clarke-Emerson[12] は，検査対象 (並行システム) の有限状態遷移グラフを効率的に探索することで**自動検証**を可能とした．

3.2 Models-as-TransitionSystems

検査対象の性質記述に**時相論理式**を用いた事は重要な点であったが，モデル検査が創造されたもう一つの決定的要因は，検査対象を表現している**有向グラフ (構造) を効率的に探索**する方法の考案であった．これにより，**構造が時相論理式のモデルになっているかどうか**を効率的に決定できるようになった．そして検証の研究には，

演繹的手法に加えてモデル的手法も登場して歴史的転換となった．

モデル検査の専門家にとっては“ クリプキ構造 ”イコール“ 状態遷移系 ”と思われるが，Models-as-TransitionSystems, Frames-as-Automata(“ フレーム ”イコール“ オートマトン ”), Worlds-as-States(“ 可能世界 ”イコール“ 状態 ”) の観点は自明ではないと考えられる．ここで，モデル検査の“ モデル ”は，論理の意味論における**構造**，及び検証における検査対象の**模型**の二重の意味を持っていると解釈できる．Models-as-TransitionSystems の観点は，論理と計算の世界を繋ぐもう一つの側面となっている．

モデル検査一般については，文献 [16] が詳しい．確率的モデル検査ツール PRISM[17] のサイトにはモデル検査ツールのリスト [18] がまとめられている．ここには，MRMC(Markov Reward Model Checker)[19], 統計的モデル検査ツール PLASMA Lab[20] など最近の情報も含まれている．

4 まとめ

Hoare 論理の誕生からモデル検査が創造された 1980 年前後の転換点までを中心にながめてきたが，その後における theory から practice への発展は目覚ましい．いわゆるモデル検査を中心にした検証の歴史，背景，展望については，Clarke[21], Emerson[22]，[23] の文献が興味深い．さらには，定理証明とモデル検査との協調 [24]；高階モデル検査 [25]；分離論理 (Separation logic)[26, 3]；性能評価，定量評価を可能にする確率的モデル検査，確率的シミュレーション，統計的検査技法などの重要な要素技術も注目に値する．

参考文献

[1] C. A. R. Hoare: An Axiomatic Basis for Computer Programming, Commun. ACM 12 (10), pp. 576–580, 1969.

[2] X. Leroy: Formal Verification of a Realistic Compiler, Commun. ACM 52 (7), pp. 107–115, 2009.

[3] G. Klein, *et. al.*: seL4: Formal Verification of an Operating-System Kernel, Commun. ACM 53 (7), pp. 107–115, 2010.

[4] Coq: https://coq.inria.fr/

[5] Isabelle: http://isabelle.in.tum.de/index.html

[6] Y. Bertot, P. Castéran: *Interactive Theorem Proving and Program Development. Coq'Art: The Calculus of Inductive Constructions*, Springer, 2004.

[7] K. Fujita, A. Schubert: The undedicability of type related problems in the type-free style System F with finitely stratified polymrphic types, Information and Computations 218, pp. 69–87, 2012.

[8] 藤田憲悦: ラムダ計算の型問題について–数学基礎論からプログラミング言語の構造へ–，数学 66-1, pp. 78–89, 2014.

[9] P. Wadler: Propositions as Types, Commun. ACM 58 (12), pp. 75–84, 2015.

[10] F. Wiedijk: The Seventeen Prvers of the World, http://www.cs.ru.nl/~freek/comparison/comparison.pdf

[11] 永宮，藤田: http://www.cs.gunma-u.ac.jp/~fujita/nagamiya080310_html/research.htm

[12] E. M. Clarke, E. A. Emerson: Design and Synthesis of Synchronization Skeletons Using Branching Time Temporal Logic, LNCS 131, pp. 52–71, 1981.

[13] J-P. Queille, J. Sifakis: Specification and verification of concurrent systems in CESAR, LNCS 137, pp. 337–351, 1982.

[14] A. Pnueli: The Temporal Logic of Programs, FOCS, pp. 46–57, 1977.

[15] M. Ben-Ari, Z. Manna, A. Pnueli: The Temporal Logic of Branching Time, POPL, pp. 164–176, 1981.

[16] C. Baier, J-P. Katoen: *Principles of Model Checking*, The MIT press, 2007.

[17] PRISM: http://www.prismmodelchecker.org/

[18] Probabilistic Model Checking Tools: http://www.prismmodelchecker.org/other-tools.php

[19] MRMC: http://www.mrmc-tool.org/trac/

[20] PLASMA Lab: https://project.inria.fr/plasma-lab/

[21] E. M. Clarke: The Birth of Model Checking, LNCS 5000 (25 Years of Model Checking), 2008.

[22] E. A. Emerson: The beginning of Model Checking: A Personal Perspective, LNCS 5000, 2008.

[23] E. M. Clarke, E. A. Emerson, J. Sifakis: Model Checking: Algorithmic Verification and Debugging, Commun. ACM 52 (11), pp. 75–84, 2009.

[24] D. A. Peled: *Software Reliability Methods*, Springer, 2001.

[25] N. Kobayashi: Model Checking Higher-Order Programs, J. of the ACM 60-3, 62 pages, 2013.

[26] C. Calcagno, D. Distefano, P. O'Hearn, H. Yang: Compositional Shape Analysis by means of Bi-Abduction, POPL, pp. 289–399, 2009.