ソフトウェア工学 134－6
（2001．10．18）

ソフトウェア開発へのモデル検査技法の応用

NEC ネットワーキング研究所
中島 震
nakajima@ccm.cl.nec.co.jp

チュートリアル概要 開発の上流工程で早期に効率良くシステムの不具合を発見するための手段として、形式検証技術に期待が集まっている [11][15]。

形式検証技術は大きく分けて、定理証明に基づく方法とモデル検査技法と呼ばれる方法とがある [6][27]。定理証明技法は適用範囲が広い反面、ツールが提供する部分的な自動検証機能を利用者が工夫しながらシステム全体の検証を行なう。そのため、高度な数学的な知識を必要とし、実用化の障壁が高い。一方、モデル検査技法は、対象を有限の状態空間で表現し、状態空間の網羅的な探索を行なうことで種々の不具合を見つける。自動検証の技術として注目されている。

モデル検査技法の歴史は古く 1970 年代に遡ることができる。近年、モデル検査アルゴリズムの進展と共に、実用的なツール [7][13][28] が公開され、多くの成功事例が蓄積されている。研究の初期には、通信プロトコルやハードウェア順序回路など状態遷移モデルと相性の良い対象への適用が多かった。ハードウェアについては実用段階に達し、例えば、インテル社ではモデル検査技術に基づく設計ルールチェックツールを日常的に使っている [10]。最近はモデル検査検証技法をソフトウェア開発に応用する試みが増えている。

実際、通信プロトコルや分散アルゴリズム [12]、要求仕様 [2][3] や分析設計過程の支援 [5][9]、ファイルシステムのキャッシュプロトコル [32]、ワークフロー [18] あるいは業務フロー [23]、ソフトウエアアーキテクチャ [29] の振舞い仕様 [1][20]、交換機の高度サービス [14]、コンポーネント基盤 [24]、セキュリティプロトコル [30]、ネットワークシステムのサバイバビリティ [17]、EC システム [19][31]、Java プログラム [8]、等に適用した事例が報告されている。また、より「軽量な (light-weighted)」使い方を目指し、状態モデルとその検証技術を Java 並行プログラミングの教育 [21] や設計 [26] に用いる試みもある。さらに、UML[34] 等のダイアグラムベースのモデリング言語との統合も興味深い [25]。たとえば、Alcoa[16] はクラス図相当のモデリング記法 Alloy の自動検証ツールである。

モデル検査検証技法をソフトウェアに応用した事例を具体的に紹介する。以下の例では、G. Holzmann (ベル研) が開発し公開しているオートマトンベースのモデル検査ツール SPIN [4][13] を用いる。E. Clarke (CMU) が公開している SMV [7] と同様にツール開発者以外の利用実績が多い。

SPIN では、Promela と呼ぶチャネル通信オートマトンを表現する言語を用いて検証対象システムを記述する。Promela 記述を Buchi オートマトンに変換し、デッドロックや無限ループの有無をチェックする。また、LTL (Linear Temporal Logic) [22] を用いて、システムが満たすべき性質を表現することも可能である。LTL の式は Buchi オートマトンに変換・検証される。SPIN/Promela の具体例として、簡単な例題 [12] と、Enterprise JavaBeans コンポーネント基盤の検証事例 [24] を紹介する。後者は、実用的なソフトウェアデザインへ適用して不具合を発見した成功事例のひとつである。

最後に、モデル検査技法に関する情報源をまとめる。ソフトウェア開発への適用事例は、IEEE/ACM ICSE、ACM SIGSOFT FSE、FME、等の国際学会で発表されることが多い。特に、2001 年 5 月にトロントで開催された ICSE 2001 では、モデル検査技法を利用した研究の発表が 8 件もあった。なお、ホームページ [33] は、モデル検査検証技法を含むフォーマルメソッドの研究分野を知る上で便利である。SPIN や SMV を含む多くのモデル検査ツールの URL も掲載されている。

参考文献

[1] R. Allen, D. Garlan, and J. Ivers : Formal Modeling and Analysis of the HLA Component Integration Standard, Proc. ACM SIGSOFT FSE'98, pp.70-79 (1998).

[2] R. Anderson, P. Beame, S. Burns, W. Chan, F. Modugno, D. Notkin, and J. Reese : Model Checking Large Software Specifications, Proc. SIGSOFT'96, pp.156–166 (1996).

[3] J. Atlee and J. Gannon : State-Based Model Checking of Event-Driven System Requirements, IEEE trans. SE, Vol.19, No.1, pp.24–40 (1993).

[4] K. Bang, J. Choi, and C. Yoo : Comments on "The Model Checker SPIN", IEEE trans. SE, Vol.27, No.6, pp.573–576 (2001).

[5] M. Chechik and J. Gannon : Automatic Analysis of Consistency between Requirements and Design, IEEE trans. SE, Vol.27, No.7, pp.651–672 (2001).

[6] E. Clarke and J. Wing : Formal Methods: State of the Art and Future Directions, ACM Computing Surveys, Vol.28, No.4, pp.626–643 (1996).

[7] E. Clarke, O. Grumberg, and D. Peled : *Model Checking*, The MIT Press 1999.

[8] E. Corbett, M. Dwyer, J. Hatcliff, S. Laubach, C. Pasareanu, Robby, and H. Zheng : Bandera – Extracting Finite-State Models from Java Source Code, Proc. ICSE 2000 (2000).

[9] S. Easterbrook and M. Chechik : A Framework for Multi-Valued Reasoning over Inconsisntent Viewpoints, Proc. ICSE 2001, pp.411–420 (2001).

[10] R. Gerth : Model Checking if Your Life Depends on It: A View from Intel's Trenches, invited talk at SPIN 2001 (2001).

[11] 萩谷 昌己, 米崎 直樹 : ソフトウェア発展と検証技術の未来, bit, Vol.32, No.12, pp.3–8 共立出版 (2000).

[12] G. Holzmann : *Design and Validation of Computer Protocols*, Prentice Hall 1991. 水野, 東野, 佐藤, 太田 (共訳): コンピュータプロトコルの設計法, カットシステム (1994).

[13] G. Holzmann : The Model Checker SPIN, IEEE trans. SE, Vol.23, No.5, pp.279–295 (1997).

[14] G. Holzmann and M. Smith: A Practical Method for Verifying Event-Driven Software, Proc. ICSE'99, pp.597–608 (1999).

[15] G. Holzmann : Economics of Software Verification, Proc. ACM SIGPLAN/SIGSOFT PASTE'01, pp.80–85 (2001).

[16] D. Jackson : Alcoa: The Alloy Constraint Analyzer, Proc. ICSE 2000, pp.730–733 (2000).

[17] S. Jha and J. Wing : Survivability Analysis of Networked Systems, Proc. ICSE 2001, pp.307–317 (2001).

[18] C. Karamanolis, D. Giannakopoulou, J. Magee, and S. Wheater : Model Checking of Workflow Schemas, Proc. EDOC 2000, pp.170-181 (2000).

[19] Y. Kesten, A. Klein, A. Pnueli, and G. Rannan : A Perfecto Verification: Combining Model Checking with Deductive Analysis to Verify Real-Life Software, Proc. FME FM'99, pp.173–194, LNCS 1708, Springer-Verlag (1999).

[20] J. Magee, J. Kramer, and D. Giannakopoulou : Analysing the Behaviour of Distributed Software Architectures: a Case Study, Proc. IEEE FTDCS'97 (1997).

[21] J. Magee and J. Kramer : *Concurrency – State Model & Java Programs*, Wiley 1999.

[22] Z. Manna and A. Pnueli : *The Temporal Logic of Reactive and Concurrent Systems : Specification*, Springer-Verlag 1991.

[23] 中島 震 : CCS を用いた業務フロー図の検証, 日本ソフトウェア科学会 FOSE'94 予稿集 (1994).

[24] S. Nakajima and T. Tamai : Behavioural Analysis of the Enterprise JavaBeans Component Architecture, Proc. SPIN 2001, pp. 163–182, LNCS 2057, Springer-Verlag (2001).

[25] 中島 震 : オブジェクト指向デザインと形式手法 (チュートリアル), コンピュータソフトウェア, Vol.18, No.5, pp.17–46 (2001).

[26] 中島 震, 前田 直人 : 実行時改変可能なサービスブローカの移動エージェントによる実現, 日本ソフトウェア科学会 第 18 回大会予稿集 (2001).

[27] A. Pnueli : Deduction is Forever, invited talk at FM'99 (1999).

[28] A. Roscoe : Model-checking CSP, in *A Classical Mind – Essays in Honour of C.A.R.Hoare*, pp.353–378 Prentice Hall 1994.

[29] M. Shaw and D. Garlan : *Software Architecture*, Prentice Hall 1996.

[30] 田中 慎也, 佐藤 文明, 水野 忠則 : SPIN に基づくセキュリティプロトコル検証システム, 情報処理学会論文誌, Vol. 42, No. 2, pp.147–154 (2001).

[31] W. Wang, Z. Hidvegi, A. Bailey Jr., and A. Whinston : E-Process Design and Assurance Using Model Checking, IEEE Computer, pp.48–53 (October 2000).

[32] J. Wing and M. Vaziri-Farahan : Model Checking Software Systems: A Case Study, Proc. ACM SIGSOFT '95, pp.128–139 (1995).

[33] `http://www.afm.sbu.ac.uk/`

[34] `http://www.omg.org/uml/`