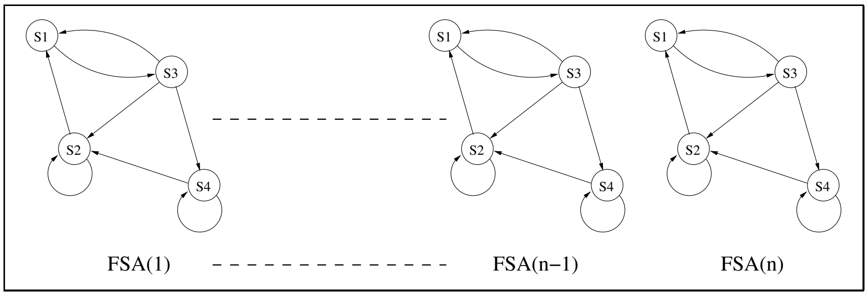
Deductive Verification of Infinite Systems: Broadcast Protocols

Michael Fisher and Alexei Lisitsa

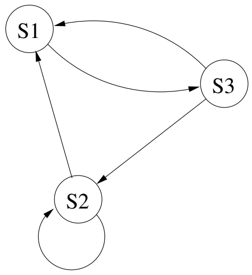
Department of Computer Science, University of Liverpool, Liverpool L69, U.K.
{M.Fisher,A.Lisitsa}@csc.liv.ac.uk

The predominant approach to the automatic verification of finite-state systems is *model checking* [1, 13]. When we consider the more general case of infinite state systems, semi-automated techniques such as tactic-based theorem-proving [17, 16] are often used. However, in recent years, significant research has been carried out extending model checking techniques to infinite state systems [5, 15], thus allowing the possibility of automatic verification of certain properties of such systems. Much work has been carried out on devising restricted classes of systems which, though having infinite state space, can also admit decidable model checking problems [8]. A particular class of system that has attracted much attention are broadcast systems comprising an arbitrary number of processes [6, 4]. Such protocols have a number of practical uses, including modelling of cache coherence protocols [3], multicast network protocols [7] and multi-agent systems [9].

Broadcast Protocols The basic idea with such protocols is to have a set of identical finite state automata executing synchronously, and communicating using simple broadcast: if one automaton makes a broadcast action, then it is requried that *all* other automata simultaneously make a message receipt reaction. This captures the model of instantaneous, synchronous broadcast communication. The infinite nature of the problem comes about since we can have arbitrary numbers of such processes in a broadcast system. Consequently, the aim is to verify that some property holds of the systems *regardless* of how many processes take part.



Specification in First-Order Temporal Logic In order to verify a broadcast protocol, we translate the problem into first-order temporal logic and then apply deductive techniques. Recall that simple finite state automata can often be concisely described using propositional temporal logic [18]. So, we can use a propositional temporal formula to represent a single process in a broadcast system. For example, if we have a proposition In_s for every state s in the finite state automaton, then formulae such as $In_{s_1} \Rightarrow \bigcirc In_{s_2}$ describe a transition within the finite state automaton, for example:



$$\begin{array}{lcl} In_{S1} & \Rightarrow & \bigcirc In_{S3} \\ In_{S2} & \Rightarrow & \bigcirc (In_{S1} \vee In_{S2}) \\ In_{S1} & \Rightarrow & \bigcirc (In_{S1} \vee In_{S2}) \end{array}$$

Now, when we consider an arbitrary number of identical finite state automata all we do is parameterise the temporal propositions by the name of the automaton being described. Thus, "$In_{s_1}(a) \Rightarrow \bigcirc In_{s_2}(a)$" means that if automaton 'a' is at state s_1 now, it will move to state s_2 in the next moment. Since we can have variables, as well as constants, then we can assert $\forall X.\ In_{s_1}(X) \Rightarrow \bigcirc In_{s_2}(X)$ meaning that *every* automaton moves from state s_1 to state s_2. In this way, we can build up a first-order temporal description of a broadcast system.

Research Questions Our initial work on this approach [11, 10] has considered the verification of restricted classes of broadcast system using deductive techniques for *monodic* first-order temporal logics [12, 2]. Our current work is looking at a range of further work, including: practical use of monodic temporal resolution provers [14] in verifying non-trivial systems; the verification of *asynchonous* broadcast protocols; more detailed analysis of the varieties of broadcast systems and their corresponding sub-classes of first-order temporal logics.

References

[1] E.M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, December 1999.

[2] A. Degtyarev, M. Fisher, and B. Konev. Monodic Temporal Resolution. In *Proc. CADE-19*, LNAI. Springer, 2003. Available as a technical report from `http://www.csc.liv.ac.uk/research/`.

[3] G. Delzanno. Automatic Verification of Parameterized Cache Coherence Protocols. In *Computer Aided Verification*, pages 53–68, 2000.

[4] G. Delzanno, J. Esparza, and A. Podelski. Constraint-Based Analysis of Broadcast Protocols. In *Proc. Annual Conf. of the European Association for Computer Science Logic (CSL'99)*, volume 1683 of *LNCS*, pages 50–66. Springer-Verlag, 1999.

[5] E. A. Emerson and K. S. Namjoshi. On Model Checking for Non-Deterministic Infinite-State Systems. In *Logic in Computer Science*, pages 70–80, 1998.

[6] J. Esparza, A. Finkel, and R. Mayr. On the Verification of Broadcast Protocols. In *Proc. 14th IEEE Symp. Logic in Computer Science (LICS '99), Trento, Italy, July 1999*, pages 352–359. IEEE Comp. Soc. Press, 1999.

[7] J. Esparza and M. Maidl. Simple Representative Instantiations for Multicast Protocols. In H. Garavel and J. Hatcliff, editors, *Proc. of TACAS '03*, number 2619 in Lecture Notes in Computer Science, pages 128–143. Springer-Verlag, 2003.

[8] J. Esparza. Decidability of Model Checking for Infinite-State Concurrent Systems. *Acta Informatica*, 34(2):85–107, 1997.

[9] M. Fisher. Characterising Simple Negotiation as Distributed Agent-Based Theorem-Proving. In *Proceedings of the Fifth International Conference on Multi-Agent Systems (ICMAS)*, Boston, USA, 2000.

[10] M. Fisher and A. Lisitsa. Temporal Deductive Verification of Cache Coherence Protocols, 2003. Submitted for journal publication; available as a technical report from `http://www.csc.liv.ac.uk/research/`.

[11] M. Fisher and A. Lisitsa. Deductive Verification of Cache Coherence Protocols. In *Proceedings of 3rd International Workshop on Automated Verification of Critical Systems (AVoCS 2003)*, pages 177–186, Southampton, UK, 2003.

[12] I. Hodkinson, F. Wolter, and M. Zakharyashev. Decidable Fragments of First-Order Temporal Logics. *Annals of Pure and Applied Logic*, 2000.

[13] G. J. Holzmann. *The Spin Model Checker: Primer and Reference Manual*. Addison-Wesley, November 2003.

[14] U. Hustadt and B. Konev. TRP++ 2.0: A Temporal Resolution Prover. In *Proc. CADE-19*, *LNAI*, Springer-Verlag, 2003. Available as a technical report from `http://www.csc.liv.ac.uk/research/`

[15] M. Leuschel and T. Massart. Infinite State Model Checking by Abstract Interpretation and Program Specialisation. In *Logic-Based Program Synthesis and Transformation (LOPSTR'99)*, volume 1817 of *LNCS*, pages 63–82. Springer-Verlag, 2000.

[16] S. Owre, J. Rushby, N. Shankar, and D. Stringer-Calvert. PVS: an experience report. In D. Hutter, W. Stephan, P. Traverso, and M. Ullman, editors, *Applied Formal Methods—FM-Trends 98*, volume 1641, pages 338–345, Boppard, Germany, 1998. Springer-Verlag.

[17] L. C. Paulson. *A Generic Theorem Prover*, volume 828 of *LNCS*. Springer-Verlag: Heidelberg, Germany, 1994.

[18] S. Safra and M. Y. Vardi. On ω-Automata and Temporal Logic. In *STOC*, pages 127–137, Seattle, Washington, May 1989. ACM.