Monodic Abstract State Machines and Temporal Verification

Alexei Lisitsa and Michael Fisher
Department of Computer Science, University of Liverpool, Liverpool L69 7ZF, U.K.
email:{A.Lisitsa, M.Fisher}@csc.liv.ac.uk

The aim of this work is to provide a mechanism for verifying Abstract State Machines by translating them to an appropriate fragment of first-order temporal logic.

Abstract State Machines. The underlying idea which provides Abstract State Machines (ASMs) with much of their flexibility, is that states of an arbitrary computation are represented by general *many-sorted first-order structures*. The vocabularies of these structures may be freely chosen in order to model any particular algorithm at an appropriate level of abstraction [10]. (This flexibility contrasts with many standard models, such as Turing machines or finite state automata, where the alphabets used are fixed.) The process of computation is then modelled by the evolution of states according to given rules; on being 'fired', these rules change the structure produced by re-interpreting its basic functions.

Since its introduction, the ASM approach has proved to be very successful in specifying a variety of different types of software and hardware, including abstract algorithms, realistic programming languages, and distributed computations [1]. The combination of generality, varying levels of abstractness and strong formal foundations has led to the ASM approach beginning to be applied in significant industrial applications [11].

Deductive temporal verification of ASM. In our ongoing work [8, 9], we pursue the goal of automatic deductive verification for certain classes of ASM. In particular, we base our work on a translation of general ASMs to full first-order temporal logic with natural numbers as a time flow [6]. While such a logic is, in general, not finitely axiomatisable, recent work has identified a fragment, termed the *monodic* fragment, that is finitely axiomatisable [12]. Furthermore, some of its natural (e.g two variables or monadic) subfragments, are decidable.

We define a class of *monodic ASMs* whose semantics in terms of temporal logic fits within the monodic fragment. This, together with recent work on clausal resolution methods [7] for monodic fragments [3, 13, 4], allows us to carry out temporal verification of properties of monodic ASMs. The approach is illustrated by the deductive verification of the correctness of both a FloodSet algorithm[14] for the Consensus problem (in the presence of crash failures) and a cache coherence protocol [5] both specified using monodic ASMs.

Research Questions and Further Work. We have shown how to faithfully translate ASM specifications into the first-order temporal logic (FOTL) and have defined restrictions on ASM specifications (programs) which ensure that temporal translation is in a 'good' (monodic) fragment of FOTL. This allows us to use temporal translations for (semi-)automatic verifications of restricted ASM specifications either by decision procedures [4], or by theorem proving for restricted fragments of FOTL [2]. However the current translation procedure tends to produce large FOTL formulae. Our current work is looking at possible optimizations of the translation, especially concerning so called "frame conditions"[8, 9]. Design of effective automatic algorithms, where possible, is an important future development, as is an in-depth analysis of the applicability of monodic ASMs.

References

[1] Abstract State Machines: A Formal Method for Specification and Verification. [`http://www.eecs.umich.edu/gasm/`].

[2] J. Brotherston, A. Degtyarev, M. Fisher, and A. Lisitsa. Implementing Invariant Search via Temporal Resolution. In *Proc. LPAR-2002*. Springer Verlag, 2002.

[3] A. Degtyarev and M. Fisher. Towards First-Order Temporal Resolution. In *Proc. KI-2001*, pages 18–32. Springer Verlag, 2001. Volume 2174 of *LNAI*.

[4] A. Degtyarev, M. Fisher, and B. Konev. Monodic temporal resolution. Proc. CADE'03, Springer.

[5] G. Delzanno, Constraint-based verification in Parametrized Cache Coherence Protocols, to appear in Formal Methods in System Design

[6] E. A. Emerson. Temporal and Modal Logic. In *Handbook of Theoretical Computer Science*, pages 996–1072. Elsevier, 1990.

[7] M. Fisher, C. Dixon, and M. Peim. Clausal Temporal Resolution. *ACM Transactions on Computational Logic*, 2(1):12–56, Jan. 2001.

[8] M. Fisher, A. Lisitsa. Temporal Verification of Monodic Abstract State Machines Technical Report ULCS-03-011, University of Liverpool Department of Computer Science, April 2003, avaialable at `http://www.csc.liv.ac.uk/research/techreports`

[9] M. Fisher, A. Lisitsa. Monodic ASMs and Temporal Verification In *Proceedings of Abstract State Machines Workshop ASM 2004*, May 2004, to appear

[10] Y. Gurevich. Evolving Algebras 1993: Lipari Guide. In *Specification and Validation Methods*, pages 9–36. Oxford University Press, 1995.

[11] Y. Gurevich and W. Schulte and M. Veanes. Toward Industrial Strength Abstract State Machines. Technical Report MSR-TR-2001-98, Microsoft Research, October 2001.

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

[13] B. Konev, A. Degtyarev, C. Dixon, M. Fisher, and U. Hustadt. Towards the Implementation of First-Order Temporal Resolution: the Expanding Domain Case. In *Proceedings of TIME-ICTL 2003,* July 2003, Cairns, Queensland, Australia. IEEE CS Press.

[14] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, San Mateo, CA, 1996.