Hidden Algebra and Concurrent Distributed Software

Joseph Goguen
University of California at San Diego
e-mail: goguen@cs.ucsd.edu

Introduction Cleverly designed software often fails to strictly satisfy its specifications, but instead satisfies them *behaviorally*, in the sense that they *appear* to be true under all possible experiments. *Hidden algebra* extends prior work on abstract data types and algebraic specification [2, 6] to concurrent distributed systems, in a surprisingly simple way that also handles nondeterminism, internal states, and more [4, 3]. Advantages of an algebraic approach include decidability results in equational logic for problems that are undecidable for more expressive logics, and powerful algorithms like term rewriting and unification, for implementing equational logic tools. Much work in formal methods has addressed code verification, but since empirical studies show that little of software cost comes from coding errors, our approach focuses on behavioral specification and verification at the design level, thus avoiding the distracting complications of programming language semantics.

Theory Hidden algebra uses *behavioral satisfaction*, whereby equations *appear* to be satisfied under all possible experiments; this forces a slight restriction of the inference rules for equational logic to preserve soundness. Our most significant results are powerful coinduction proof rules, which can greatly reduce proof size compared with more classical methods [3, 9]. Term rewriting has also been extended to hidden algebra, and integrated with coinduction [8], to support a high degree of proof automation for behavioral properties, which in general may be first order sentences with equations as atoms. Hidden algebra can be seen as a generalization of process algebra, transition systems, and coalgebra.

Practice The practical side of this project is developing a web-based distributed cooperative environment for behavioral specification and verification, and applying it to concurrent distributed systems such as protocols. The main component is *Kumo*, a proof assistant that facilitates browsing and understanding by publishing its proofs on the web, integrated with explanations and background tutorials [1]. Kumo uses the *BOBJ* specification language, which extends OBJ3 [7, 2] with behavioral features, and the *Duck* proof scripting language. There is also a database for managing projects, specs, proofs and users. The distributed character of the system requires a coherence protocol, which has been verified with hidden algebra [5]. Kumo is available for experimentation at `http://www.cs.ucsd.edu/groups/tatami/kumodb.html`; see also the Kumo homepage, `http://www.cs.ucsd.edu/groups/tatami/kumo/`.

References

[1] Joseph Goguen, Kai Lin, Akira Mori, Grigore Roşu, and Akiyoshi Sato. Distributed cooperative formal methods tools. In Michael Lowry, editor, *Proceedings, Automated Software Engineering*, pages 55–62. IEEE, 1997.

[2] Joseph Goguen and Grant Malcolm. *Algebraic Semantics of Imperative Programs*. MIT, 1996.

[3] Joseph Goguen and Grant Malcolm. Hidden coinduction: Behavioral correctness proofs for objects. *Mathematical Structures in Computer Science*, 9(3):287–319, June 1999.

[4] Joseph Goguen and Grant Malcolm. A hidden agenda. *Theoretical Computer Science*, to appear. Also UCSD Dept. Computer Science & Eng. Technical Report CS97–538, May 1997.

[5] Joseph Goguen and Grigore Roşu. A protocol for distributed cooperative work. In Gheorghe Stefanescu, editor, *Proceedings, FCT'99, Workshop on Distributed Systems*, pages 1–22. Elsevier, 1999. (Iaşi, Romania). Also, Electronic Lecture Notes in Theoretical Computer Science, Elsevier Volume 28, to appear 1999.

[6] Joseph Goguen, James Thatcher, and Eric Wagner. An initial algebra approach to the specification, correctness and implementation of abstract data types. In Raymond Yeh, editor, *Current Trends in Programming Methodology, IV*, pages 80–149. Prentice-Hall, 1978.

[7] Joseph Goguen, Timothy Winkler, José Meseguer, Kokichi Futatsugi, and Jean-Pierre Jouannaud. Introducing OBJ. In Joseph Goguen and Grant Malcolm, editors, *Software Engineering with OBJ: Algebraic Specification in Action*. Kluwer, to appear. Also Technical Report SRI-CSL-88-9, August 1988, SRI International.

[8] Grigore Roşu. Behavioral coinductive rewriting. In Kokichi Futatsugi, Joseph Goguen, and José Meseguer, editors, *OBJ/CafeOBJ/Maude at Formal Methods '99*, pages 179–196. Theta (Bucharest), 1999. Proceedings of a workshop in Toulouse, 20 and 22 September 1999.

[9] Grigore Roşu and Joseph Goguen. Hidden congruent deduction. In Ricardo Caferra and Gernot Salzer, editors, *Proceedings, 1998 Workshop on First Order Theorem Proving*, pages 213–223. Technische Universität Wien, 1998. Full version to appear, *Lecture Notes in Artificial Intelligence*, Springer, 1999.