Panel: Logic in the Computer Science Curriculum*

Kim B. Bruce
Williams College
Williamstown, MA 01267
email: kim@cs.williams.edu

Phokion G. Kolaitis
University of California
Santa Cruz, CA 95064
email: kolaitis@cse.ucsc.edu

Daniel M. Leivant
Indiana University
Bloomington, IN 47405
email: leivant@cs.indiana.edu

Moshe Y. Vardi (Moderator)
Rice University
Houston, TX 77005-1892,
email: vardi@cs.rice.edu

Logic has been called "the calculus of computer science" [1]. The argument is that logic plays a fundamental role in computer science, similar to that played by calculus in the physical sciences and traditional engineering disciplines. Unlike calculus, however, the central place of logic in the computer science curriculum is far from universally accepted. For example, the ACM/IEEE Computing Curricula 1991 lists logic only as one of many mathematics requirements. Yet logic plays an important role in areas of Computer Science as disparate as architecture (logic gates), software engineering (specification and verification), programming languages (semantics, logic programming), database (relational algebra and SQL), artificial intelligence (automatic theorem proving), algorithms (complexity and expressiveness), and theory of computation (general notions of computability). Some might even argue that much of computer science can be seen as a generalization or outgrowth of logic. On the other hand, there are those who claim that logic is just an academic exercise, of no practical import.

The focus of this panel the role of logic in the computer science curriculum. The panelists addressed the following questions:

1. What is the content of logic in CS? Is it the standard package of propositional logic and first-order logic or should we include modal logic, temporal logic, and the like?
2. Should Logic in CS be taught differently from Logic in Math? In what way? Are the Association for Symbolic Logic (ASL) Guidelines for Teaching Logic, published in Volume 1 of the Bulletin of the ASL, 1995, relevant to computer science?
3. Where does logic belong in the CS curriculum: lower-division courses? upper division courses? graduate courses? all of the above? Should logic be a required or an optional part of the curriculum?
4. How does logic fit with the rest of the CS curriculum? What courses should it require? What courses should require it?

*This panel resulted from several discussions that took place during the 1995-6 DIMACS Special Year on Logic and Algorithms. DIMACS is an NSF-funded Science and Technology Center for Discrete Mathematics and Theoretical Computer Science and one of the New Jersey Commission on Science and Technology's Advanced Technology Centers. See http://dimacs.rutgers.edu.

Kim Bruce

It is interesting to note how the requirements for logic-related material have changed in the successive Computer Science curriculum standards. Neither Curriculum '68 or Curriculum '78 placed much emphasis on logic aside from an introduction to propositional (and perhaps predicate) logic in a discrete structures course. Curricula '91 and the liberal arts CS curriculum recommendations, however, increased the amount of attention to logic related areas (including material traditionally placed in a theory of computation course).

I like to think of logic as a (relatively) simple formal language that can be used to introduce computer scientists to interesting issues in programming languages and algorithms. How should these issues be packaged? Hopefully some of it will come in a Discrete Mathematics course taught in a student's first year of college. However, I find that mathematics departments often shy away from the level of theory and proof that we find necessary. On the other hand, CS curricula are already too filled with material, with more banging on the door to get into the undergraduate curriculum. As a result, I believe that most of

these topics need to be integrated into existing courses. In CS we have for too long ghettoized theory into a separate course which has little impact on the rest of the curriculum. These ideas should be introduced in intro, data structures, and computer organization courses, with further polishing and depth introduced in algorithms, theory, programming languages, and other courses (e.g., database, compilers, etc.) During the presentation I will propose concrete logic-related topics to be inserted in these courses.

Phokion G. Kolaitis

In an ideal world where resources are plentiful there should be an upper division undergraduate course on logic in computer science. In reality, however, most CS departments can not afford to have such a course in their undergraduate curriculum. What is to be done, then? There are plenty of opportunities to cover a fair amount of logic in several undergraduate courses (both lower division and upper division ones) and tie this material to concrete applications. Here are three such courses:

- *Introduction to CS*: In a general introduction to CS for non-majors or pre-majors, one can cover the basics of propositional logic and relate it to gates, circuit design, etc.
- *Discrete Mathematics*: A thorough treatment of propositional logic should be presented in this course.
- *Database Systems*: This course provides an excellent opportunity to present the basics of first-order logic in the guise of relational calculus and show how SQL is essentially a first-order language.

Daniel M. Leivant

Computer Science is the first science intimately linked to logic. The incorporation of logic into the computer science curriculum should therefore be conceived independently of traditional pedagogical approaches (notably that of mathematical logic).

Logic has been a branch of philosophy for millenia, and became a branch of mathematics in the 19th century, with impressive foundational results but relatively small impact on mathematical practice. In contrast, central notions of computer science are part and parcel of logic: abstraction and encapsulation, syntactic entities as objects of discourse, precise syntactic rules and operations, and the dichotomy of syntax and semantics. Indeed, computer science should be viewed as a branch of applied logic, and computer scientists are, for the most part unknowingly, applied logicians.

Consequently, logic should be presented to computer scientists as the conceptual underpinning of computer science itself. Using traditional mathematical logic courses, supplemented with minor computer-science related afterthoughts, misses the mark and merely alienates those students who have no particular interest in mathematical logic as such.

Logic should be present in the CS curriculum at several levels:

1. Certain undergraduate courses should have a well conceived logic component.
2. An elective undergraduate course of logic in computer science (LICS) should be available.
3. A required or strongly recommended introductory graduate course in LICS should be available.
4. Clearly conceived logic components should be present in certain graduate level courses, and these should already assume familiarity with the logic fundamentals (programming language theory, semantics, AI, databases)

Moshe Y. Vardi

I believe that logic should be an integral part of the computer curriculum. The formalisms of propositional and first-order logic should be taught early in the curriculum (perhaps as part of discrete math) and be relied on in more advanced courses, e.g., hardware courses should rely on propositional logics, databases courses should rely on first-order logic, and the like. At the upper division level, an advanced course in logic, covering the basic concepts of propositional and first-order logic (truth, validity, provability, etc.) should be offered as an elective.

Logic in computer science should be taught, however, from the computer science perspective. Thus, there needs to be a strong emphasis on algorithmic aspects. For example, unique readability should be taught also as a parsing issue, propositional satisfiability should be connected to NP-completeness, provability should be viewed as polynomial-time checkability, and undecidability should be emphasized over incompleteness. For lectures notes that describe how logic in computer science is taught at Rice university, see `http://www.cs.rice.edu/~vardi/comp409`.

References

[1] MANNA, Z. AND WALDINGER, R. *The Logical Basis for Computer Programming*. Addison-Wesley, 1985.