Review of **Logic for Computer Scientists**[*]

Riccardo Pucella

Department of Computer Science
Cornell University

July 29, 2004

This is a short introductory book on the topic of propositional and first-order logic, with a bias towards computer scientists. What's a bias towards computer scientists, you ask? Good question. Despite my initial belief, it does not mean that it introduces logic with an eye towards the wide variety of areas where logic is used in computer science. Rather, it gives a computational perspective on logic; the emphasis is on computational aspects of logic, and specifically on procedures for establishing the satisfiability or unsatisfiability of formulas, culminating in various forms of the resolution procedure. (In that same sense, Smullyan [5] gives a tableaux-based perspective on first-order logic.) The study of resolution procedures leads naturally to the topic of using predicate logic as a programming language, that is, logic programming.

This focus on computational aspects means that there is much less coverage of those topics typically found in logic textbooks, such as axiomatization and completeness results, or model theoretic notions such as applications of compactness and theories. This is not a criticism, mind you: there are other good introductory books that deal with that. Schöning decides to concentrate on computational issues, and gives us a short book (less than 170 pages) with a tight storyline.

Summary of content. The book consists of three chapters.

Chapter 1 is about propositional logic. After the obligatory introduction to the syntax and semantics of propositional logic, the two standard normal forms for formulas are presented, namely disjunctive and conjunctive normal forms. The special class of Horn formulas is introduced, and an efficient algorithm for deciding satisfiability of Horn formulas is given. The compactness of propositional logic is then established: a set of formulas is satisfiable if and only if every finite subset is satisfiable. Building on this result, the resolution calculus for propositional is defined: a set of rules that can be applied to clause sets (an alternate representation for a formula in conjunctive normal form) to derive new clauses. The resolution calculus is proved to be refutation complete: if the empty clause is derivable in the calculus, then the original clause set is unsatisfiable, and if the original clause set is unsatisfiable, then the empty clause is derivable. This gives a procedure for deciding satisfiability of propositional formulas. (A specialization of the resolution calculus to Horn formulas, unit resolution, appears in Exercise 35.)

Chapter 2 is about first-order logic. After the presentation the syntax and semantics of first-order logic, normal forms for first-order formulas are introduced: prenex form (where all quantifiers are in the front of the formula), Skolem form (where all existential quantifier have been removed and all existentially quantified variables replaced by a function of the remaining variables), and a clause set representation extending that of propositional logic. The undecidability of the validity problem for first-order logic is proved by reduction from the Post correspondence problem. (The decidability of monadic predicate logic appears in Exercise 69.) A quick mention of theories is made, but not pursued. The important class of term models (also known as Herbrand structures) is introduced, and used (in passing) to prove the Löwenheim-Skolem theorem. The Herbrand expansion of a first-order formula, which produces an infinite propositional clause set by replacing every variable of the formula in its clause set representation by an element of

[*] U. Schöning, *Logic for Computer Scientists*, Birkhäuser, 1989, 166pp, ISBN 0817634533.

the term model, is defined, and compactness of propositional logic is used to prove Herbrand's theorem: a first-order formula is unsatisfiable if and only if there exists a finite subset of its Herbrand expansion that is unsatisfiable. As an application of Herbrand's theorem, a procedure due to Gilmore that enumerates the clauses of the Herbrand expansion of a formula and thus provides a semi-decidable procedure for the satisfiability problem of first-order logic is presented. Gilmore's procedure is then modified to use propositional resolution as its main step, yielding the ground resolution procedure for first-order logic. After analyzing the difficulties with this procedure, the resolution procedure of Robinson is described; it works directly on first-order clause sets, and requires unification of first-order terms. This resolution procedure is shown to be complete for first-order formulas. A numbers of refinements of the general procedure are presented, along with completeness results; these include unit resolution, which is defined for all formulas, but is complete for (first-order) Horn formulas, and SLD resolution, which is defined only for Horn formulas, and is also complete for them.

Chapter 3 is about logic programming. More precisely, it studies the problem of turning first-order logic into a programming language. The idea is well known. Start with a *query* formula of the form $\exists x.P(t,x)$, where t is a variable-free term. Clearly, this formula is valid if and only if $\forall x.\neg P(t,x)$ is unsatisfiable. A resolution refutation of $\forall x.\neg P(t,x)$ showing it unsatisfiable can be modified to yield a c such that $P(t,c)$ holds; this c is called an *answer* to the query. In this way, it is possible to view first-order logic as a query language driven by the resolution procedure. For efficiency reasons, this language is often restricted to Horn formulas, with appropriate restrictions to the resolution procedure (such as using SLD resolution), and using further special evaluation strategies to resolve the nondeterminism inherent in the resolution procedure. The programming language Prolog is described; it is probably the best known practical language that uses Horn formulas with a restricted resolution procedure.

Opinion. I found this a nicely written book with many examples and exercises (126 of them). The presentation is natural and easy to follow. It should be noted, however, that due to both its length, and its date of publication (the first printing of the English edition was in 1989), recent developments in logic programming are not included. Since this book is meant to be introductory, it is fair to ask how it relates to other "popular" introductory books on logic. Well, what do I have on my bookshelf that compares? Let's leave aside the two books from which I learned logic, Enderton [1] and Shoenfield [4], which are both much more mathematical. A book that is much closer in spirit is Nerode and Shore [3], who also in some sense target computer scientists. There is an overlap in topics, as Nerode and Shore also discuss resolution, and much more. Schöning's book holds up surprisingly well to Nerode and Shore's, in no small part due to it being shorter and much more focussed. Finally, going back to my initial belief prompted by the title of the book, a good textbook that describes the uses of logic in computer science, with an emphasis on verification, is that of Huth and Ryan [2].

This book seems suitable for a short course, a seminar series, or part of a larger course on Prolog and logic programming, probably at the advanced undergraduate level. For a general logic course, even in computer science, it should be supplemented to give a more complete picture of the subject.

References

[1] H. B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 1972.

[2] M. Huth and M. Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge University Press, 1999.

[3] A. Nerode and R. Shore. *Logic for Applications*. Springer-Verlag, 1994.

[4] J. R. Shoenfield. *Mathematical Logic*. Addison Wesley, 1967.

[5] R. Smullyan. *First-Order Logic*. Springer-Verlag, 1968.