Jean H. Gallier. **Logic for computer science. Foundations of automatic theorem proving.** Computer science and technology series. Harper & Row, New York.

Logic for computer science is a senior-level undergraduate textbook that covers an unusual slice of the fields of mathematical logic and automated theorem proving. This slice includes soundness and completeness of a Gentzen system for first-order logic, Gentzen's sharpened Hauptsatz and Herbrand's theorem, resolution, logic programming, and congruence closure. All these topics are centered around Gentzen systems which makes the book coherent and cohesive.

Some topics one might expect in a modern textbook on logic for computer science which are not covered are intuitionistic logic, higher-order logic, and Gödel's incompleteness theorems. Their absence is justified in that no textbook can reasonably be expected to cover the ever-growing intersection between logic and computer science.

The main strengths of **Logic for computer science** are its lucid presentation and mathematical rigor that clearly exposes the relationship between different topics. It uses proof-theoretic and semantics methods, both of which a Computer Science student should be familiar with. All too often, proof-theoretic methods are neglected in favor of shorter, and superficially more elegant semantic arguments. The formulation of the inference systems and proofs of metatheorems are often original, thus containing much material of interest even to readers versed in the field. This does not distract from its value as a textbook, which is enhanced further by a multitude of good exercises at all levels of difficulty.

Occasionally the author follows too many closely related developments, and some of the material might profitably be omitted in favor of more extensive coverage of applications. This concentration on foundations that the author owns up to in the subtitle is perhaps the only weakness of **Logic for computer science** as an undergraduate textbook.

After reviewing basic mathematical tools, Chapter 3 begins the study of logic by giving a thorough treatment of propositional logic. This encompasses syntax, semantics, and some proof theory for a Gentzen system. Many important notions are introduced and illustrated in this simple setting that are ordinarily reserved for first-order logic. This includes König's Lemma, Hintikka sets, consistency, compactness, and cut-elimination.

Chapter 4 then presents a resolution procedure for propositional logic. Its

soundness and completeness is proved through explicit translations between resolution refutations and Gentzen proofs. Nice syntactic arguments of this kind pervade the book and should be very accessible to students. A point made repeatedly is that transformational proofs of this sort also provide more information—in this case about the complexity of resolution proofs compared to the length of Gentzen proofs.

Chapter 5 generalizes from propositional logic to first-order logic. The treatment of the proof theory of the Gentzen system is oriented towards computation with proofs. For example, a pseudo-Pascal version of a complete search procedure for first-order cut-free Gentzen proofs is presented. The Löwenheim-Skolem and compactness theorems and axiomatizations of arithmetic are only briefly discussed and perhaps warrant some more detail. An interesting topic that is usually not discussed in much depth is the addition of equality to first-order logic, and precisely how the results obtained for an equality-free logic extend to this case.

Chapter 6 develops Gentzen's cut-elimination theorem and presents some of its applications. Again, the approach is syntactic and a version of Gentzen's Hauptsatz is proved constructively, including the bound on the size of a proof resulting from cut-elimination. The value of proof transformations and syntactic methods over semantic and generally uninformative proofs is demonstrated very lucidly. Transformational methods pervade Computer Science and it is very obvious that this book is written by a Computer Scientist and for students of Computer Science. The author also gives an alternative semantic completeness argument, thus contrasting semantical and proof-theoretic arguments.

In Chapter 7 the cut-elimination theorem is further refined into Herbrand's theorem by way of Gentzen's sharpened Hauptsatz. The proof of Herbrand's theorem is again explicit through proof transformations.

The results from Chapter 7 are put to good use in Chapter 8, where a version of resolution for first-order logic is presented. Applications of resolution beyond general theorem proving are then discussed in Chapter 9 in which an idealized Prolog is introduced and related to resolution. Unfortunately, no real programming language is given. I believe that owning up to the differences between the logical basis for logic programming and a real Prolog would significantly strengthen this chapter and would give students an opportunity for hands-on experience with the connection between logic and logic programming. Still, this chapter and Chapter 10 on many-sorted

languages and congruence closure are a very important part of **Logic for computer science**, since they connect the largely mathematical exposition of first-order logic in the other chapters to applications in Computer Science.

In summary, I highly recommend **Logic for computer science** as a textbook for a senior-level undergraduate Computer Science course. Its well-structured and rigorous development of basic concepts in logic specifically directed towards students in Computer Science should make it a very successful basis for such a course.

— Frank Pfenning