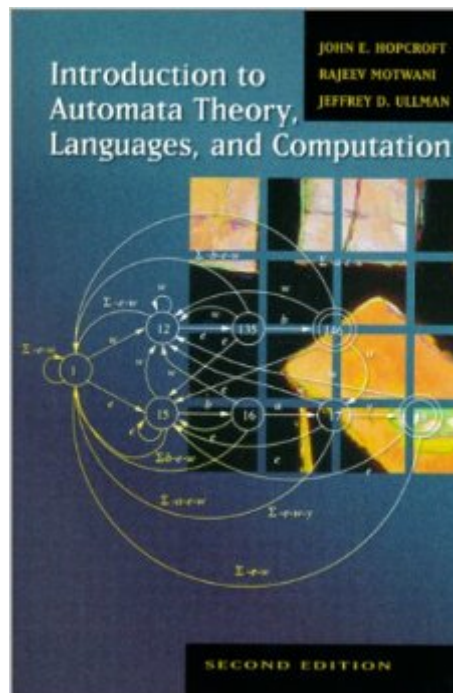# Introduction To Automata Theory, Languages, And Computation (2nd Edition)

Synopsis

It has been more than 20 years since this classic book on formal languages, automata theory, and computational complexity was first published. With this long-awaited revision, the authors continue to present the theory in a concise and straightforward manner, now with an eye out for the practical applications. They have revised this book to make it more accessible to today's students, including the addition of more material on writing proofs, more figures and pictures to convey ideas, side-boxes to highlight other interesting material, and a less formal writing style. Exercises at the end of each chapter, including some new, easier exercises, help readers confirm and enhance their understanding of the material.

Book Information

Hardcover: 521 pages

Publisher: Addison Wesley; 2 edition (November 24, 2000)

Language: English

ISBN-10: 0201441241

ISBN-13: 978-0201441246

Product Dimensions: 6.4 x 1.3 x 9.6 inches

Shipping Weight: 1.6 pounds

Average Customer Review: 3.7 out of 5 stars See all reviews (69 customer reviews)

Best Sellers Rank: #355,434 in Books (See Top 100 in Books)   #57 in Books > Computers & Technology > Computer Science > AI & Machine Learning > Machine Theory   #115 in Books > Science & Math > Mathematics > Pure Mathematics > Number Theory   #123 in Books > Science & Math > Mathematics > Pure Mathematics > Discrete Mathematics

Customer Reviews

This is a good book - but as a revision of a much-revered classic of the field, it's a bit of a disappointment. Hopcroft & Ullman wrote the classic text way back in 1969, and then revised it in 1979. It was pretty much the standard text the world over for an introduction to the theory of computation. But over the last two decades, more and more people have been studying Computer science, and many of them have no time for theory and formalism and all the 'dry stuff' .... The authors point out that because of such reasons and also because nowadays there's little research in the theory of computation per se, and more in its applications, they've written a book to cater to today's students. Which, in other words, means they've simplified the presentation, tried to provide intuition whenever possible, given lots more examples and done away with some of the more

difficult material. This approach puts the book into direct competition with Michael Sipser's excellent 'Introduction to the theory of computation', a contest it cannot win, though it might be a respectable second. Almost all topics are motivated by giving examples of how they're related to applications in the 'real world', and similar to Sipser's 'proof idea' approach, the authors first present a topic informally and then formally, thus gently leading the reader to the formal proofs.

I've just passed my exam on Theory of Computation, and I've used both editions of this text. Frankly speaking, I couldn't choose one of the two should I keep only one of them.Whereas the first was full of strict formalism, the second has traded this for a more discursive approach. Whereas the first reported theorems name (of their authors), the second has traded this for a richer bibliography at the end of the chapters. And more objectively, the first edition covered more "classical" topics with shorter treatments than the second, but this last treats survived topics with richer details (starting from the first chapter on mathematical basis for the course) and with updated examples of applications (XML and Markup Languages, e-commerce for DFA, etc).This said, you know why I can't decide. A discursive approach is of course always desiderable, especially if you're completely new to a subject, but a strong notation is helpful in my mind because it improves communication and removes ambiguities. Hence, the best approach would probably have been a mix of the two, or halfway the two.As a second matter, having a rich bibliography is surely helpful both for further studies and as a reference, but it's quite tedious to look at the index and be unable to find something like "Kleene theorem": you've to dive into bibligraphy to discover that "L is an L(DFA) if and only if it also is L(REG)" is something that has been studied by Kleene.

Introduction to Automata Theory, Languages, and Computation (2nd Edition) Introduction to Automata Theory, Languages, and Computation (3rd Edition) Introduction to Automata Theory, Languages, and Computation Cellular Automata: 8th International Conference on Cellular Automata for Research and Industry, ACRI 2008, Yokohama, Japan, September 23-26, 2008, Proceedings (Lecture Notes in Computer Science) The Global Dynamics of Cellular Automata: An Atlas of Basin of Attraction Fields of One-Dimensional Cellular Automata (Santa Fe Institute Studies in the Sciences of Complexity Reference Volumes) Introduction to Languages and the Theory of Computation Formal Languages and Their Relation to Automata (Addison-Wesley Series in Computer Science and Information Processing) The Languages of Tolkien's Middle-Earth: A Complete Guide to All Fourteen of the Languages Tolkien Invented Introduction to the Theory of Computation Using MPI - 2nd Edition: Portable Parallel Programming with the Message Passing

Interface (Scientific and Engineering Computation) Finite Automata and Regular Expressions: Problems and Solutions The Wonderland Of Music Boxes And Automata Amazing Automata -- Dinosaurs! (Dover Origami Papercraft) Paper Models That Rock!: Six Pendulum Automata (Dover Origami Papercraft) Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning series) Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning series) Introduction to Machine Learning (Adaptive Computation and Machine Learning series) Common LISP: A Gentle Introduction to Symbolic Computation (Dover Books on Engineering) Seizing the Light: A Social History of Photography 2nd (second) Edition by Hirsch, Robert published by McGraw-Hill Humanities/Social Sciences/Languages (2008) Masterminds of Programming: Conversations with the Creators of Major Programming Languages (Theory in Practice (O'Reilly))