

# The Craft Of Prolog Logic Programming

The Quest for Artificial Intelligence  
The Reasoned Schemer  
Logic for Computer Scientists  
Functional and Logic Programming  
The Craft of Prolog  
Learn Prolog Now!  
Logic for Applications  
Declarative Logic Programming  
Next Generation Information System Technology  
Logic for Computer Science  
Adventure in Prolog  
Logic, Programming and Prolog  
The Art of Prolog  
Logic Programming  
Micro-PROLOG  
Programming with Higher-Order Logic  
Simply Logical  
Logic Programming  
Logic for Problem Solving  
Thinking As Computation  
Logic Programming  
An Introduction to Logic Programming Through Prolog  
Logic, Programming and Prolog  
Techniques of Prolog Programming with Implementation of Logical Negation and Quantified Goals  
From Logic Programming to Prolog  
Logic Programming  
Computational Logic and Human Thinking  
Parallel Logic Programming  
Clause and Effect  
Answer Set Programming  
Computing with Logic  
Prolog: Programming For Artificial Intelligence, 3/E  
Concepts in Programming Languages  
Multiagent Systems  
The Practice of Prolog  
The Art and Logic of Ramon Llull  
Logic Programming with Prolog  
Parallel Logic Programming  
Functional and Logic Programming  
Prolog Programming in Depth

## The Quest for Artificial Intelligence

This volume contains the proceedings of two Russian conferences on logic programming, held in 1990 in Irkutsk and in 1991 in St. Petersburg. The aim of the conferences was to bring together researchers from the Russian and the international logic programming communities. Topics covered in the volume include: logic programming, automated theorem proving, non-monotonic reasoning, applications of mathematical logic to computer science, deductivedatabases, implementation of declarative concepts, and programming in non-classical logics. This is the first volume to represent activity in the field of logic programming in the countries of the former Soviet Union.

## The Reasoned Schemer

Provides a systematic introduction to the theory of logic programming and shows how this theory can be applied to reason about pure Prolog programs. The text includes an introduction to programming in Prolog and deals with such programming issues as determination, occur-check freedom and absence of errors. It covers both the natural interpretations of logic programming, as declarative specification and as procedure for computer execution.

## Logic for Computer Scientists

A comprehensive undergraduate textbook covering both theory and practical design issues, with an emphasis on object-oriented languages.

## Functional and Logic Programming

The idea of this book grew out of a symposium that was held at Stony Brook in September 2012 in celebration of David S. Warren's fundamental contributions to

Computer Science and the area of Logic Programming in particular. Logic Programming (LP) is at the nexus of Knowledge Representation, Artificial Intelligence, Mathematical Logic, Databases, and Programming Languages. It is fascinating and intellectually stimulating due to the fundamental interplay among theory, systems, and applications brought about by logic. Logic programs are more declarative in the sense that they strive to be logical specifications of "what" to do rather than "how" to do it, and thus they are high-level and easier to understand and maintain. Yet, without being given an actual algorithm, LP systems implement the logical specifications automatically. Several books cover the basics of LP but focus mostly on the Prolog language with its incomplete control strategy and non-logical features. At the same time, there is generally a lack of accessible yet comprehensive collections of articles covering the key aspects in declarative LP. These aspects include, among others, well-founded vs. stable model semantics for negation, constraints, object-oriented LP, updates, probabilistic LP, and evaluation methods, including top-down vs. bottom-up, and tabling. For systems, the situation is even less satisfactory, lacking accessible literature that can help train the new crop of developers, practitioners, and researchers. There are a few guides on Warren's Abstract Machine (WAM), which underlies most implementations of Prolog, but very little exists on what is needed for constructing a state-of-the-art declarative LP inference engine. Contrast this with the literature on, say, Compilers, where one can first study a book on the general principles and algorithms and then dive in the particulars of a specific compiler. Such resources greatly facilitate the ability to start making meaningful contributions quickly. There is also a dearth of articles about systems that support truly declarative languages, especially those that tie into first-order logic, mathematical programming, and constraint solving. LP helps solve challenging problems in a wide range of application areas, but in-depth analysis of their connection with LP language abstractions and LP implementation methods is lacking. Also, rare are surveys of challenging application areas of LP, such as Bioinformatics, Natural Language Processing, Verification, and Planning. The goal of this book is to help fill in the previously mentioned void in the LP literature. It offers a number of overviews on key aspects of LP that are suitable for researchers and practitioners as well as graduate students. The following chapters in theory, systems, and applications of LP are included.

### **The Craft of Prolog**

Written for those who wish to learn Prolog as a powerful software development tool, but do not necessarily have any background in logic or AI. Includes a full glossary of the technical terms and self-assessment exercises.

### **Learn Prolog Now!**

The emphasis in The Craft of Prolog is on using Prolog effectively. It presents a loose collection of topics that build on and elaborate concepts learned in a first course.

### **Logic for Applications**

This text covers natural language processing in Prolog and presumes knowledge of Prolog, but not of linguistics. It includes simple but practical database query systems; covers syntax, formal semantics, and morphology; emphasizes working computer programs that implement subsystems of a natural language processor; features programs that are clearly designed and compatible with any Edinburgh-compatible prolog implementation (Quintas, ESL, Arity, ALS etc.); and contains nearly 100 hands-on Prolog programming exercises and problem sets.

### **Declarative Logic Programming**

This second edition contains revised chapters taking into account recent research advances. More advanced exercises have been included, and "Part II The Prolog Language" has been modified to be compatible with the new Prolog standard. This is a graduate level text that can be used for self-study.

### **Next Generation Information System Technology**

This book constitutes the refereed proceedings of the 5th International Symposium on Functional and Logic Programming, FLOPS 2001, held in Tokyo, Japan in March 2001. The 21 revised full papers presented together with three invited papers were carefully reviewed and selected from 40 submissions. The book offers topical sections on functional programming, logic programming, functional logic programming, types, program analysis and transformation, and Lambda calculus.

### **Logic for Computer Science**

OCTOBER 29 - NOVEMBER 1, 1990, AUSTIN, TEXAS Theory and Foundations. Metaprogramming. Constraints. Implementations, Architecture. Deductive Databases. Language Issues. Relation to Other Paradigms. Parallelism, Concurrency. Compilation Techniques. Applications.

### **Adventure in Prolog**

Prolog is a programming language, but a rather unusual one. Prolog" is short for "Programming with Logic", and the link with logic gives Prolog its special character. At the heart of Prolog lies a surprising idea: don't tell the computer what to do. Instead, describe situations of interest, and compute by asking questions. Prolog will logically deduce new facts about the situations and give its deductions back to us as answers. Why learn Prolog? For a start, its "say what the problem is, rather than how to solve it" stance, means that it is a very high level language, good for knowledge rich applications such as artificial intelligence, natural language processing, and the semantic web. So by studying Prolog, you gain insight into how sophisticated tasks can be handled computationally. Moreover, Prolog requires a different mindset. You have to learn to see problems from a new perspective, declaratively rather than procedurally. Acquiring this mindset, and learning to appreciate the links between logic and programming, makes the study of Prolog both challenging and rewarding. Learn Prolog Now! is a practical introduction to this fascinating language. Freely available as a web-book since 2002 (see [www.learnprolognow.org](http://www.learnprolognow.org)) Learn Prolog Now! has become one of the most popular

introductions to the Prolog programming language, an introduction prized for its clarity and down-to-earth approach. It is widely used as a textbook at university departments around the world, and even more widely used for self study. College Publications is proud to present here the first hard-copy version of this online classic. Carefully revised in the light of reader's feedback, and now with answers to all the exercises, here you will find the essential material required to help you learn Prolog now.

### **Logic, Programming and Prolog**

In writing this book, our goal was to produce a text suitable for a first course in mathematical logic more attuned than the traditional textbooks to the recent dramatic growth in the applications of logic to computer science. Thus, our choice of topics has been heavily influenced by such applications. Of course, we cover the basic traditional topics: syntax, semantics, soundness, completeness and compactness as well as a few more advanced results such as the theorems of Skolem-Lowenheim and Herbrand. Much of our book, however, deals with other less traditional topics. Resolution theorem proving plays a major role in our treatment of logic especially in its application to Logic Programming and PROLOG. We deal extensively with the mathematical foundations of all three of these subjects. In addition, we include two chapters on nonclassical logics - modal and intuitionistic - that are becoming increasingly important in computer science. We develop the basic material on the syntax and semantics (via Kripke frames) for each of these logics. In both cases, our approach to formal proofs, soundness and completeness uses modifications of the same tableau method introduced for classical logic. We indicate how it can easily be adapted to various other special types of modal logics. A number of more advanced topics (including nonmonotonic logic) are also briefly introduced both in the nonclassical logic chapters and in the material on Logic Programming and PROLOG.

### **The Art of Prolog**

This book attempts to explain the functioning of the combinatorial, semi-mechanical demonstrative techniques of Ramon Llull's 'Art', how it began as an apologetic instrument, how it developed through two main stages, and how it ended trying to reformulate key aspects of medieval Aristotelian logic.

### **Logic Programming**

The goal of this book is to show the beauty and elegance of relational programming, which captures the essence of logic programming. The book shows how to implement a relational programming language in Scheme, or in any other functional language, and demonstrates the remarkable flexibility of the resulting relational programs

### **Micro-PROLOG**

Highly parallel machines have been available for many years but, because advances in hardware have always outpaced progress in software development,

designers and users of these machines have yet to realize their full potential. Until recently there have been few, if any, high-class parallel programming languages that could be implemented on the wide variety of parallel processing systems in use. This book helps to redress the balance by teaching programming techniques as well as performance analysis of parallel programming languages and architectures using logic programming; specifically, it focuses on the Prolog-like languages OR-parallel Prolog and AND-parallel FGHC. Parallel Logic Programming brings to light practical applications of a previously esoteric/theoretical area of parallel logic programming and is unique in presenting programming hand-in-hand with performance analysis of real empirical measurements. Its quantitative approach to symbolic parallel programming provides students and professionals with tools for implementing and critically evaluating larger projects. The book includes useful chapter summaries, programming projects, and a glossary. Evan Tick is Assistant Professor in the Department of Computer Science at the University of Oregon.

### **Programming with Higher-Order Logic**

The papers in this book discuss the concepts that will determine the next generation of information systems, such as data types for databases, object orientation, data deduction and construction, visual database interfaces, interoperability and extensibility, open architectures.

### **Simply Logical**

Highly parallel machines have been available for many years but, because advances in hardware have always outpaced progress in software development, designers and users of these machines have yet to realize their full potential. Until recently there have been few, if any, high-class parallel programming languages that could be implemented on the wide variety of parallel processing systems in use. This book helps to redress the balance by teaching programming techniques as well as performance analysis of parallel programming languages and architectures using logic programming; specifically, it focuses on the Prolog-like languages OR-parallel Prolog and AND-parallel FGHC. Parallel Logic Programming brings to light practical applications of a previously esoteric/theoretical area of parallel logic programming and is unique in presenting programming hand-in-hand with performance analysis of real empirical measurements. Its quantitative approach to symbolic parallel programming provides students and professionals with tools for implementing and critically evaluating larger projects. The book includes useful chapter summaries, programming projects, and a glossary. Evan Tick is Assistant Professor in the Department of Computer Science at the University of Oregon.

### **Logic Programming**

This book introduces the notions and methods of formal logic from a computer science standpoint, covering propositional logic, predicate logic, and foundations of logic programming. The classic text is replete with illustrative examples and exercises. It presents applications and themes of computer science research such

as resolution, automated deduction, and logic programming in a rigorous but readable way. The style and scope of the work, rounded out by the inclusion of exercises, make this an excellent textbook for an advanced undergraduate course in logic for computer scientists.

### **Logic for Problem Solving**

Answer set programming (ASP) is a programming methodology oriented towards combinatorial search problems. In such a problem, the goal is to find a solution among a large but finite number of possibilities. The idea of ASP came from research on artificial intelligence and computational logic. ASP is a form of declarative programming: an ASP program describes what is counted as a solution to the problem, but does not specify an algorithm for solving it. Search is performed by sophisticated software systems called answer set solvers. Combinatorial search problems often arise in science and technology, and ASP has found applications in diverse areas—in historical linguistic, in bioinformatics, in robotics, in space exploration, in oil and gas industry, and many others. The importance of this programming method was recognized by the Association for the Advancement of Artificial Intelligence in 2016, when AI Magazine published a special issue on answer set programming. The book will introduce the reader to the theory and practice of ASP. It will describe the input language of the answer set solver CLINGO, which was designed at the University of Potsdam in Germany and is used today by ASP programmers in many countries. It will include numerous examples of ASP programs and present the mathematical theory that ASP is based on. There will be many exercises with complete solutions.

### **Thinking As Computation**

This book is for people who have done some programming, either in Prolog or in a language other than Prolog, and who can find their way around a reference manual. The emphasis of this book is on a simplified and disciplined methodology for discerning the mathematical structures related to a problem, and then turning these structures into Prolog programs. This book is therefore not concerned about the particular features of the language nor about Prolog programming skills or techniques in general. A relatively pure subset of Prolog is used, which includes the 'cut', but no input/output, no assert/retract, no syntactic extensions such as if then-else and grammar rules, and hardly any built-in predicates apart from arithmetic operations. I trust that practitioners of Prolog programming who have a particular interest in the finer details of syntactic style and language features will understand my purposes in not discussing these matters. The presentation, which I believe is novel for a Prolog programming text, is in terms of an outline of basic concepts interleaved with worksheets. The idea is that worksheets are rather like musical exercises. Carefully graduated in scope, each worksheet introduces only a limited number of new ideas, and gives some guidance for practising them. The principles introduced in the worksheets are then applied to extended examples in the form of case studies.

### **Logic Programming**

"This volume contains the proceedings of two Russian conferences on logic programming, held in 1990 in Irkutsk and in 1991 in St. Petersburg. The aim of the conferences was to bring together researchers from the Russian and the international logic programming communities. Topics covered in the volume include: logic programming, automated theorem proving, non-monotonic reasoning, applications of mathematical logic to computer science, deductivedatabases, implementation of declarative concepts, and programming in non-classical logics. This is the first volume to represent activity in the field of logic programming in the countries of the former Soviet Union."--PUBLISHER'S WEBSITE.

## **An Introduction to Logic Programming Through Prolog**

### **Logic, Programming and Prolog**

Multiagent systems combine multiple autonomous entities, each having diverging interests or different information. This overview of the field offers a computer science perspective, but also draws on ideas from game theory, economics, operations research, logic, philosophy and linguistics. It will serve as a reference for researchers in each of these fields, and be used as a text for advanced undergraduate or graduate courses. The authors emphasize foundations to create a broad and rigorous treatment of their subject, with thorough presentations of distributed problem solving, game theory, multiagent communication and learning, social choice, mechanism design, auctions, cooperative game theory, and modal logics of knowledge and belief. For each topic, basic concepts are introduced, examples are given, proofs of key results are offered, and algorithmic considerations are examined. An appendix covers background material in probability theory, classical logic, Markov decision processes and mathematical programming.

### **Techniques of Prolog Programming with Implementation of Logical Negation and Quantified Goals**

This advanced text for undergraduate and graduate students introduces mathematical logic with an emphasis on proof theory and procedures for algorithmic construction of formal proofs. The self-contained treatment is also useful for computer scientists and mathematically inclined readers interested in the formalization of proofs and basics of automatic theorem proving. Topics include propositional logic and its resolution, first-order logic, Gentzen's cut elimination theorem and applications, and Gentzen's sharpened Hauptsatz and Herbrand's theorem. Additional subjects include resolution in first-order logic; SLD-resolution, logic programming, and the foundations of PROLOG; and many-sorted first-order logic. Numerous problems appear throughout the book, and two Appendixes provide practical background information.

### **From Logic Programming to Prolog**

### **Logic Programming**

Formal systems that describe computations over syntactic structures occur frequently in computer science. Logic programming provides a natural framework for encoding and animating such systems. However, these systems often embody variable binding, a notion that must be treated carefully at a computational level. This book aims to show that a programming language based on a simply typed version of higher-order logic provides an elegant, declarative means for providing such a treatment. Three broad topics are covered in pursuit of this goal. First, a proof-theoretic framework that supports a general view of logic programming is identified. Second, an actual language called  $\lambda$ Prolog is developed by applying this view to higher-order logic. Finally, a methodology for programming with specifications is exposed by showing how several computations over formal objects such as logical formulas, functional programs, and  $\lambda$ -terms and  $\pi$ -calculus expressions can be encoded in  $\lambda$ Prolog.

### **Computational Logic and Human Thinking**

This new edition covers the latest advances in logic programming research. Its clear and authoritative treatment of the theory is combined with an emphasis on practical programming issues. This is a broad and highly readable introduction to the subject for advanced students and programmers.

### **Parallel Logic Programming**

Students explore the idea that thinking is a form of computation by learning to write simple computer programs for tasks that require thought. This book guides students through an exploration of the idea that thinking might be understood as a form of computation. Students make the connection between thinking and computing by learning to write computer programs for a variety of tasks that require thought, including solving puzzles, understanding natural language, recognizing objects in visual scenes, planning courses of action, and playing strategic games. The material is presented with minimal technicalities and is accessible to undergraduate students with no specialized knowledge or technical background beyond high school mathematics. Students use Prolog (without having to learn algorithms: "Prolog without tears"), learning to express what they need as a Prolog program and letting Prolog search for answers. After an introduction to the basic concepts, Thinking as Computation offers three chapters on Prolog, covering back-chaining, programs and queries, and how to write the sorts of Prolog programs used in the book. The book follows this with case studies of tasks that appear to require thought, then looks beyond Prolog to consider learning, explaining, and propositional reasoning. Most of the chapters conclude with short bibliographic notes and exercises. The book is based on a popular course at the University of Toronto and can be used in a variety of classroom contexts, by students ranging from first-year liberal arts undergraduates to more technically advanced computer science students.

### **Clause and Effect**

The practical benefits of computational logic need not be limited to mathematics and computing. As this book shows, ordinary people in their everyday lives can

profit from the recent advances that have been developed for artificial intelligence. The book draws upon related developments in various fields from philosophy to psychology and law. It pays special attention to the integration of logic with decision theory, and the use of logic to improve the clarity and coherence of communication in natural languages such as English. This book is essential reading for teachers and researchers who may be out of touch with the latest developments in computational logic. It will also be useful in any undergraduate course that teaches practical thinking, problem solving or communication skills. Its informal presentation makes the book accessible to readers from any background, but optional, more formal, chapters are also included for those who are more technically oriented.

### **Answer Set Programming**

Addressed to readers at different levels of programming expertise, The Practice of Prolog offers a departure from current books that focus on small programming examples requiring additional instruction in order to extend them to full programming projects. It shows how to design and organize moderate to large Prolog programs, providing a collection of eight programming projects, each with a particular application, and illustrating how a Prolog program was written to solve the application. These range from a simple learning program to designing a database for molecular biology to natural language generation from plans and stream data analysis. Leon Sterling is Associate Professor in the Department of Computer Engineering and Science at Case Western Reserve University. He is the coauthor, along with Ehud Shapiro, of The Art of Prolog. Contents: A Simple Learning Program, Richard O'Keefe. Designing a Prolog Database for Molecular Biology, Ewing Lusk, Robert Olson, Ross Overbeek, Steve Tuecke. Parallelizing a Pascal Compiler, Eran Gabber. PREDITOR: A Prolog-Based VLSI Editor, Peter B. Reintjes. Assisting Register Transfer Level Hardware Design, Paul Drongowski. Design and Implementation of a Partial Evaluation System, Arun Lakhotia, Leon Sterling. Natural Language Generation from Plans, Chris Mellish. Stream Data Analysis in Prolog, Stott Parker.

### **Computing with Logic**

1988: papers presented at the joint meeting of the Fifth International Conference on Logic Programming and the Fifth Symposium on Logic Programming.

### **Prolog: Programming For Artificial Intelligence, 3/E**

Approaches the subject by applying the format used in successful language courses. Offers a comprehensive exhibition of Prolog programming techniques in four stages--declarative, procedural, advanced and meta-programming. Presents simple and efficient implementation of logical negation and quantified goals which are necessary in expert systems. The dynamics of these new features are shown in the construction of a multilingual expert system shell that supports negative and quantified queries as well as subtypes. The easy-to-follow tutorial style and numerous fully-solved exercises facilitate understanding. Comes with 3.5 inch disk containing all programs in the book.

## **Concepts in Programming Languages**

An introduction to Prolog programming for artificial intelligence covering both basic and advanced AI material. A unique advantage to this work is the combination of AI, Prolog and Logic. Each technique is accompanied by a program implementing it. Seeks to simplify the basic concepts of logic programming. Contains exercises and authentic examples to help facilitate the understanding of difficult concepts.

## **Multiagent Systems**

Not long ago" Dennis Merritt wrote one of the best books that I know of about implementing expert systems in Prolog, and I was very glad he published it in our series. The only problem is there are still some unfortunate people around who do not know Prolog and are not sufficiently prepared either to read Merritt's book, or to use this extremely productive language, be it for knowledge-based work or even for everyday programming. Possibly this last statement may surprise you if you were under the impression that Prolog was an "artificial intelligence language" with very limited application potential. Please believe this editor's statement that quite the opposite is true: for at least four years, I have been using Prolog for every programming task in which I am given the option of choosing the language. Therefore, I 'am indeed happy that Dennis Merritt has written another good book on my language of choice, and that it meets the high standard he set with his prior book, Building Expert Systems in Prolog. All that remains for me to do is to wish you success and enjoyment when taking off on your Adventure in Prolog.

## **The Practice of Prolog**

Logic programming has increasing significance in computer science beyond the current fashion for expert systems. This book takes a software engineering rather than an expert systems/AI approach and covers logical theory, practical programming and PROLOG im

## **The Art and Logic of Ramon Llull**

Artificial intelligence (AI) is a field within computer science that is attempting to build enhanced intelligence into computer systems. This book traces the history of the subject, from the early dreams of eighteenth-century (and earlier) pioneers to the more successful work of today's AI engineers. AI is becoming more and more a part of everyone's life. The technology is already embedded in face-recognizing cameras, speech-recognition software, Internet search engines, and health-care robots, among other applications. The book's many diagrams and easy-to-understand descriptions of AI programs will help the casual reader gain an understanding of how these and other AI systems actually work. Its thorough (but unobtrusive) end-of-chapter notes containing citations to important source materials will be of great use to AI scholars and researchers. This book promises to be the definitive history of a field that has captivated the imaginations of scientists, philosophers, and writers for centuries.

## **Logic Programming with Prolog**

Basic concepts. Logic programming using micro-Prolog. Core micro-Prolog. Applications of micro-Prolog.

## **Parallel Logic Programming**

This new edition covers the latest advances in logic programming research. Its clear and authoritative treatment of the theory is combined with an emphasis on practical programming issues. This is a broad and highly readable introduction to the subject for advanced students and programmers.

## **Functional and Logic Programming**

Investigates the application of logic to problem solving and computer programming. Requires no previous knowledge in this field, and therefore can be used as an introduction to logic, the theory of problem-solving and computer programming. Annotation copyrighted by Book News, Inc., Portland, OR

## **Prolog Programming in Depth**

[ROMANCE](#) [ACTION & ADVENTURE](#) [MYSTERY & THRILLER](#) [BIOGRAPHIES & HISTORY](#) [CHILDREN'S](#) [YOUNG ADULT](#) [FANTASY](#) [HISTORICAL FICTION](#) [HORROR](#) [LITERARY FICTION](#) [NON-FICTION](#) [SCIENCE FICTION](#)