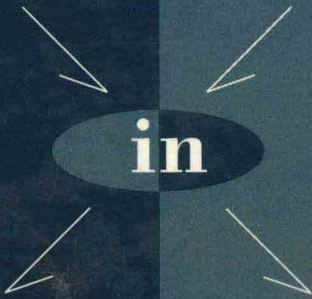


Artificial
Intelligence



in

Chemical
Engineering

Thomas E. Quantrille & Y. A. Liu

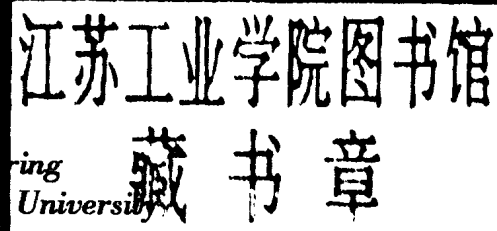
Artificial Intelligence in Chemical Engineering

Thomas E. Quantrille

and

Y. A. Liu

*Department of Chemical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA*



ACADEMIC PRESS

Harcourt Brace Jovanovich, Publishers

San Diego New York Boston London Sydney Tokyo Toronto

Academic Press Rapid Manuscript Reproduction

This book is printed on acid-free paper. ∞

Copyright © 1991 by ACADEMIC PRESS, INC.

All Rights Reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the publisher.

Academic Press, Inc.

San Diego, California 92101

United Kingdom Edition published by

Academic Press Limited

24-28 Oval Road, London NW1 7DX

Library of Congress Cataloging-in-Publication Data

Quantrille, Thomas E.

Artificial intelligence in chemical engineering / by Thomas E.

Quantrille and Y. A. Liu.

p. cm.

Includes bibliographical references and index.

ISBN 0-12-569550-0

1. Chemical engineering--Data processing. 2. Artificial intelligence--Industrial applications. I. Liu, Y. A. (Yih-An)
II. Title.

TP184.Q36 1991

660'.0285'63--dc20

91-34293

CIP

PRINTED IN THE UNITED STATES OF AMERICA

92 93 94 95 9 8 7 6 5 4 3 2 1

PREFACE

This book is intended as an introduction to artificial intelligence (AI) for chemical engineers. In the classroom, the book can serve as the textbook for a one-semester course in artificial intelligence in chemical engineering for seniors or graduate students. The book is also appropriate as a self-study guide for practicing engineers who wish to understand more about the field of AI in engineering. Finally, computer scientists interested in engineering applications of AI should find this book informative, and rich in references.

Computer applications in chemical engineering have increased dramatically over the last decade. AI techniques in particular have received much attention. This book is a response to that growth, and therefore, has two major purposes:

- To introduce chemical engineers to the principles and applications of AI.
- To introduce chemical engineers to computer programming techniques for AI.

Before we can accomplish these goals, however, we must first address a more fundamental question: *what is artificial intelligence?* There is no universally accepted definition of AI, so we cannot provide a single, clear-cut answer. But we can suggest several definitions relevant to this text. The goal of AI has always been to make computers “think,” to solve problems requiring human intelligence. Based on this idea, Elaine Rich and Kevin Knight, in their book *Artificial Intelligence* (McGraw-Hill, 1991) define AI as:

The study of how to make computers do things which, at the moment, people do better.

Avron Barr and Edward A. Feigenbaum in *The Handbook of Artificial Intelligence*, Volume I (Addison-Wesley, 1981) propose the following definition:

Artificial Intelligence is the part of computer science concerned with designing intelligent computer systems, that is, systems that exhibit characteristics we associate with intelligence in human behavior.

We prefer this latter definition; the focus of this book is to introduce the computational means and methodologies to enable computers to perform intelligent engineering tasks.

Thus, one of the primary goals of this book is to move beyond the basic principles of AI solidly into its applications in chemical engineering. After reading this book, a chemical engineer will have a firm grounding in AI, know what chemical engineering applications of AI exist today, and understand the current challenges facing AI in engineering. Importantly, after reading this book, the chemical engineer will also be able to implement AI systems using the language Prolog.

To achieve our objectives, we have divided this book into four parts:

- Introduction to AI programming, including Prolog (chapters 1 to 8) and LISP (chapter 9),

- and comparison of Prolog with LISP (chapter 9).
- Introduction to AI principles (chapters 10 and 11).
- Chemical engineering case study: the development of EXSEP, an EXpert System for SEparation Synthesis (chapters 12 to 15).
- Overview of knowledge-based systems and artificial neural networks in chemical engineering (chapters 16 and 17).

Prolog is one of the two principal languages of artificial intelligence, is used extensively in Europe and Japan, and with increasing frequency in the United States. Prolog originated in the early 1970's, where researchers attempted to use logic as a programming language. The name itself stands for PROgramming in LOGic. Robert Kowalski and Maarten van Emden established the theoretical foundations for Prolog at Edinburgh. Alain Colmerauer, at the University of Aix-Marseille in Marseille, France, completed the first implementation of Prolog. Prolog owes its current popularity to an efficient implementation credited to David Warren at Edinburgh in the mid-1970's.

Prolog is a young, "up-and-coming" language. Consequently, it has no "standard" syntax. For this book, we have chosen the most popular Prolog syntax in use today, known as the Edinburgh syntax (also called DEC-10 syntax). For the most part, we have used Arity/Prolog to develop the programs presented in this text. Arity/Prolog's syntax closely resembles Edinburgh syntax. However, we have tried to avoid using features unique to Arity/Prolog to keep the programs as broadly applicable as possible. We also provide some programs in Turbo Prolog (presently being sold under the name PDC Prolog). Turbo Prolog is less powerful than Arity/Prolog, but is also much less costly. As with Arity/Prolog, Turbo Prolog's syntax is closely related to Edinburgh syntax. Thus, applications described in this book can be applied to virtually any version of Prolog as long as differences in syntax are recognized.

Part One (chapters 1-9) introduces and discusses AI programming in detail. We discuss aspects of syntax, program control, built-in functions, and programming techniques of Prolog using chemical engineering examples. At the end of chapter 8, the reader will be able to write complex AI programs in Prolog.

Note, however, that this book is *not* strictly a Prolog book. In chapter 9, we introduce LISP (List Processing), the other major AI language, and contrast and compare Prolog and LISP. In the United States, LISP is actually used more widely than Prolog, and many chemical engineering applications of AI use LISP. LISP owes its origins to John McCarthy, who developed the language at MIT in 1959 and formally presented it in 1960. LISP is the second oldest computer language around—the oldest is FORTRAN. Because of its age, LISP has been modified and optimized more than Prolog. LISP systems generally have many more built-in functions than Prolog systems do, and in that sense, they are more powerful.

Why, then, do we emphasize Prolog? We feel that *economically*, Prolog has an advantage over LISP, when it comes to *learning and teaching AI*. We strongly feel that engineers should "get their feet wet" and understand AI *with as little time and initial financial investment as possible*. Once an engineer has a firm grasp of the field, then he or she can justify more expensive software and hardware systems if necessary. Prolog allows us to adopt this approach.

For teaching purposes:

- Prolog is simpler and easier to learn than LISP.
- Prolog provides for faster learning—an engineer can develop and implement prototype programs more quickly in Prolog than in LISP.
- Prolog programs can be developed and implemented on inexpensive personal computers. Most LISP development packages need specialized hardware, which raises the cost to both the developer and the user.
- Prolog is a relatively inexpensive investment—a user can purchase personal-computer-based Prolog software for as little as \$100.

Consequently, for learning and teaching AI systems quickly and inexpensively, we feel that Prolog is superior to LISP.

Each language, then, has its own advantages *and* disadvantages. While Prolog is faster and easier to learn, LISP is more powerful. Before choosing which language to use, the wise developer will understand these advantages and disadvantages. Indeed, Patrick Winston, Director of the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology, writes in his Foreword to the Prolog text by Ivan Bratko (Addison-Wesley, 1990, p. vii):

In the Middle Ages, knowledge of Latin and Greek was essential for all scholars. The one-language scholar was necessarily a handicapped scholar who lacked the perception that comes from seeing the world from two points of view. Similarly, today's practitioner of Artificial Intelligence is handicapped unless thoroughly familiar with both LISP and Prolog, for knowledge of the two principal languages of Artificial Intelligence is essential for a broad point of view.

Chapter 9 will help in this regard by introducing LISP and providing a comparison of both LISP and Prolog.

In Part Two (chapters 10 and 11), we discuss AI principles. Chapter 10 presents a broad overview, and discusses problem-solving techniques that have been used successfully in AI programs. We also include strategies of *knowledge representation*, i.e., systematic ways to represent knowledge (i.e., facts, rules and heuristics) such that an AI program can solve problems accurately and efficiently.

Chapter 11 is a more specific discussion of Prolog programming for AI. We introduce AI programming techniques in Prolog, including the *search*. In AI, the *search* is the methodology for *processing* the information in the program to draw conclusions and solve problems. We discuss various problem-solving approaches in AI, and discuss how to implement these approaches using Prolog. After the completion of chapter 11, the reader should have a firm grasp of Prolog and AI, and, more importantly, know how to implement Prolog programs for AI applications.

Part Three (chapters 12–15) is a chemical engineering case study. The goal of Part Three is to demonstrate, by example, how to develop, build, and implement an expert system. Chapter 12 introduces EXSEP (EXpert System for SEparation Synthesis), a process design tool for multicomponent separations in chemical engineering. EXSEP uses heuristics with shortcut feasibility analysis to develop technically feasible and economically attractive chemical process flowsheets.

Chapter 13 discusses EXSEP from the chemical engineering perspective. Through this chapter, we hope to describe how to develop a knowledge representation that opportunistically takes advantage of the chemical engineering problem characteristics. Expert-system success hinges on good knowledge representation.

Chapter 14 describes EXSEP from the user's perspective. We provide a copy of the EXSEP program listing in the Appendix, and a diskette for use on IBM PCs (or MS DOS compatible computers). We discuss EXSEP's user-interface, how to access the program, and how to interact with and use the program to develop chemical process flowsheets.

Part Three concludes with Chapter 15, where we discuss both the knowledge representation and the problem-solving strategy in EXSEP from the AI standpoint. AI researchers have developed various techniques and problem-solving strategies for expert systems. Certain techniques are more useful than others, depending on the problem. Knowing the available AI tools and techniques, as well as when to apply these tools, is critical to the success of an expert system. At the close of Part Three, the reader should know how to implement expert systems for problems of moderate to high complexity.

In Part Four (chapters 16 and 17), we "spread our wings" and take a look at recent trends in AI in science and engineering, and in chemical engineering in particular. Chapter 16 is a survey of knowledge-based applications in chemical engineering implemented in Prolog, LISP, and other AI

programming languages. We discuss how to manage expert-system projects. We then move on to specific application areas, such as fault diagnosis, process control, process design, planning and operations, modeling and simulation, and product design, development, and selection.

Finally, chapter 17 introduces a more recent AI application in chemical engineering, *Artificial Neural Networks* (ANNs). The ANN is a numerical, empirical modeling tool that has received much attention recently. ANN's origins lie in the computer modeling of the human brain, with associated attempts to emulate neuron interaction. Chapter 17 describes the unique properties that separate ANNs from other empirical modeling techniques.

Importantly, chapter 17 assumes no prior knowledge of ANNs, and discusses their purpose and structure starting with the most basic principles. We then discuss current and potential ANN applications in chemical engineering. We close the chapter with a thorough summary of the advantages *and* limitations of ANNs. Again, the wise developer will always know the limitations of any technique being used.

This text contains numerous example problems to clarify the discussion. We have included a large number of simple exercises and more complex practice problems for use as classroom assignments and self-study materials. Solutions to most of the exercises and practice problems appear in the Appendix. In addition, we include a glossary, which explains the terminology used in AI applications in science and engineering.

After working through this book, a chemical engineer will have the breadth and depth to implement AI projects using Prolog. Depending on the problem, however, LISP may be the better language choice. LISP is a more powerful language, with more built-in procedures, and thus, may be more convenient. The reader may wish to refer to *LISP*, by Patrick H. Winston and Berthold K.P. Horn (Addison-Wesley, 1984), for a discussion of the language.

Overall, this text serves as an introduction to AI techniques in engineering (and chemical engineering, in particular). It includes both a *broad coverage* and an *in-depth review* of AI techniques. We hope that this book will benefit engineering students (seniors or graduate students), practicing chemical engineers, and computer scientists interested in applications of AI to engineering.

ACKNOWLEDGEMENTS

It is a pleasure to thank a number of very special persons who contributed to the preparation of this book.

First, we would like to thank the enduring patience of our wives, Sharon Quantrille and Hing-Har Liu. Their support through the laborious process of book preparation and editing was exemplary and most appreciated.

The idea for this book originated from the doctoral work of the junior author, Tom Quantrille. The junior author would like to thank the members of his advisory committee, in particular: Professor Y. A. Liu, who developed the original idea and the details of the book and was the major advisor; Professor Peter Rony, whose thorough review and comments proved invaluable; Professor John Roach, who introduced the subject of artificial intelligence to both authors; Professor William Conger, the Department Head who provided both technical and logistic support; and finally, Professor Henry McGee, who served on the committee even while on leave at the National Science Foundation as Director of the Division of Chemical and Thermal Systems.

In addition, we wish to express our sincere gratitude to Professor Michael Mavrovouniotis of the University of Maryland and to Professor Lyle Ungar of the University of Pennsylvania for their prompt and thorough reviews of the manuscript for the whole book. Professor Mavrovouniotis was particularly helpful to our revision of chapter 17.

To Ms. Marie Paretti, we owe a deep gratitude for her tireless and skillful copy-editing of the entire manuscript for this book. We would also like to thank Ms. Lydia Morgan, editorial assistant at Academic Press, for her help in making the final pages and artwork come together and Ms. Lisa Herder of Office Arts for her electronic production of final camera ready pages.

We would also like to express our appreciation to the Dow Chemical Foundation, whose financial support made this work possible.

We wish to thank Mr. Jean-Christophe N. Brunet for his help with many aspects of this book project.

The senior author should like to thank Professor Gregory Botsaris of Tufts University for his continued support and encouragement.

Lastly and most importantly, we wish to thank Mrs. Diane Cannaday, who patiently and graciously helped all aspects of the preparation of this writing project.

RECOMMENDATIONS

This section gives some recommendations for student learning, course topics and software selection.

Student Learning and Course Topics

For a first reading through this text or for a one-semester course, we recommend that the following portions be emphasized.

- Chapters 1 to 5. These chapters give the foundation for Prolog programming for AI applications.
- In Chapters 6 to 9, focus on sections 6.1 to 6.3, 7.3, 7.5, 8.1, 8.3, 9.1 and 9.2. These sections give an understanding of some essential built-in Prolog tools, explain techniques for improving programming efficiency, and contrast Prolog with LISP. We skip over some advanced and less-used tools and techniques.
- Chapter 10. This key chapter gives the foundations of AI and expert systems.
- In Chapter 11, focus on sections 11.1 and 11.2. These sections represent the “meat” of Prolog programming for AI applications.
- In Chapters 12 to 15, focus on sections 12.2, 13.1 to 13.3, 13.6, 14.1 to 14.3, 15.1 and 15.2. These sections are the key to understanding the case study, EXSEP (EXpert system for SEparation synthesis). In addition, we recommend using the diskette for EXSEP included inside the backcover of this book to develop separation process flowsheets for examples and practice problems in section 14.4C before progressing to Chapter 15.
- Chapter 16 is a stand-alone chapter that focuses on the knowledge-based applications in chemical engineering and can be used to enhance course effectiveness if time permits. The topics discussed in sections 16.3 to 16.8 can be assigned to students to write term papers or develop course projects. In addition, students may be able to update the literature on the knowledge-based applications to specific areas by reviewing the latest issues of journals cited in these sections.
- Chapter 17 is also a stand-alone chapter. Sections 17.1 and 17.2 cover the fundamentals of artificial neural networks (ANNs), an increasingly important area of AI applications to chemical engineering. Topics discussed in section 17.3 can be assigned to students for term papers or course projects on applications of ANNs to chemical engineering.

Software Selection

We strongly recommend that the reader practice Prolog programming. There are two Prolog

packages that we recommend for IBM personal computers or compatibles under MS DOS.

1. Turbo Prolog (presently being sold under the name PDC Prolog). Prolog Development Corporation, 568 14th Street, N.W., Atlanta, GA 30318, phone (404) 873-1366. PDC Prolog is identical to Turbo Prolog, previously sold by the Borland International, Inc., Scotts Valley, CA. This Prolog software is the least expensive and the reader will need to use it to develop and compile Prolog codes, such as exercises in Part One.
2. Arity/Prolog. Arity Corporation, 29 Domino Drive, Concord, MA 01742, phone (508) 371-1243. Arity/Prolog is a powerful, well-organized, PC-based Prolog.

Both companies offer student prices and group discounts. For a class of ten or more students, Arity/Prolog is an excellent choice.

ABOUT THE AUTHORS

Thomas E. Quantrille is a Ph.D. student in the Department of Chemical Engineering, Virginia Polytechnic Institute and State University ("Virginia Tech"). He is completing his doctoral research on artificial intelligence in chemical engineering design.

Mr. Quantrille received his B.S. degree with honors from Virginia Tech in 1984. He worked for the Procter and Gamble Company as a research engineer from 1984 to 1986, and then joined James River Corporation as a technical research leader. He was granted a number of U.S. and European patents during his industrial employment. He returned to Virginia Tech in 1988 for graduate studies. For his outstanding academic progress at Virginia Tech, he has received the Pratt Presidential Award and the Dow Chemical Ph.D. Fellowship. He expects to receive his Ph.D. degree in 1991.

Mr. Quantrille's professional interests include artificial intelligence, process synthesis, and polymer science and technology. He is a member of AIChE, AAAI, and TAPPI.

Y. A. Liu is the Frank C. Vilbrandt Professor of Chemical Engineering at Virginia Tech. He received his B.S. (1967), M.S. (1970), and Ph.D. (1974) degrees from National Taiwan University, Tufts University and Princeton University, respectively.

Professor Liu taught at Auburn University from 1974 to 1981, where his last position was Alumni Associate Professor endowed by the Auburn Alumni Association. He joined Virginia Tech as a Professor of Chemical Engineering in 1982. In 1983, he was appointed the Vilbrandt Professor.

He has two patents and over 100 publications in the fields of process design, magnetothermal engineering, coal desulfurization and fluidization processes. He has also edited four books which have become standard references in these fields.

Professor Liu's contributions to chemical engineering teaching and research have been recognized by university and national awards. In 1984, he received the Western Electric Award from the American Society of Engineering Education (ASEE). The award recognizes his excellence as an engineering teacher and his contributions to the engineering profession. In 1986, Professor Liu received the National Catalyst Award from the Chemical Manufacturers Association for outstanding teaching and contributions to chemical education. In 1990, he received the George Westinghouse Award, ASEE's highest honor for an engineering educator under age 45. This award recognizes the outstanding early achievements of a young engineering educator in both teaching and scholarship. In 1990, Professor Liu also received the Distinguished Alumni Award from Tufts University.

CONTENTS

<i>Preface</i>	<i>xv</i>
<i>Acknowledgements</i>	<i>xix</i>
<i>Recommendations</i>	<i>xxi</i>
<i>About the Authors</i>	<i>xxiii</i>

PART ONE. INTRODUCTION TO AI PROGRAMMING

Chapter 1. Introduction to Prolog

1.1 Representing Facts in Prolog	3
Exercises	5
1.2 A Simple Program Including Facts and Questions	6
A. Prolog Programs	6
B. Data Objects	7
C. Matching	8
Exercises	8
1.3 An Introduction to Rules in Prolog	9
A. A Description of Rules	9
B. Using Rules in Prolog	12
Exercises	19
1.4 Recursive Rules in Prolog	20
A. Introduction to Recursion	20
B. Processing Recursive Rules in Prolog	22
Exercises	26
1.5 Chapter Summary	26
A Look Ahead	27
References	28

Chapter 2. A More In-Depth View

2.1 Data Representation and Syntax	29
A. Constants	30
B. Variables	33
C. Simple and Structured Objects	35
D. Summary	36
Exercises	37

2.2 Matching	38
A. Instantiation of Variables	38
B. Requirements for a Match	39
C. Equality	42
Exercises	43
2.3 Declarative and Procedural Nature of Prolog	43
A. Understanding Declarative and Procedural Languages	43
B. Ordering of Clauses: The Danger of Infinite Loops	44
C. Techniques in Ordering Clauses	47
Exercises	51
2.4 Prolog and Logic	52
A. Predicate Logic	52
B. Horn Clauses	53
2.5 Chapter Summary	53
A Look Ahead	54
References	54

Chapter 3. Arithmetic, Lists, and Recursion

3.1 Arithmetic in Prolog	55
A. Arithmetic Operators	56
1. Operator Notation	56
2. General Arithmetic Operators	57
3. The = and is Predicates	58
4. Mathematical Predicates	59
B. Arithmetic Comparison Operators	60
Exercises	62
3.2 Lists	63
A. Description and Syntax	63
B. Head and Tail of a List	64
C. Lists as Binary Trees	65
Exercises	66
3.3 Recursion and List Processing in Prolog	67
A. Recursion Using Lists	67
B. Comparison with Iteration	71
3.4 Practical Applications of List Processing	71
A. Append	71
B. Reverse	73
C. Add and Delete	76
D. Sublist	77
E. First and Last Elements	78
F. Length	79
G. Remove Duplicate	80
H. Summary of List Processing Relations	81
Exercises	81
3.5 Chapter Summary	84
3.6 Practice Problems	85
A Look Ahead	86
References	86

Chapter 4. Backtracking and Program Control

4.1 Preventing Backtracking: The Cut	87
4.2 Backtracking in Prolog	90
A. Understanding the Cut	90
B. Why Prolog Backtracks	94

C. How Prolog Backtracks	94
Exercises	94
4.3 Common Uses of the Cut	95
A. Making Relations Deterministic	95
B. "Locking in" a Single Rule	96
C. Negation as Failure	96
Exercises	97
4.4 Consequences of the Cut	100
A. Procedural Ordering Consequences	100
B. Goal-Matching Consequences	101
Exercises	105
4.5 Chapter Summary	106
A Look Ahead	106
References	106
 Chapter 5. Input and Output	
5.1 Reading and Writing Terms	107
A. Input and Output Streams	107
B. The write, tab, nl (new line) and read Predicates	108
5.2 Writing Lists	109
Exercises	111
5.3 Reading and Writing Characters	112
A. The put Predicate	112
B. The get and get0 Predicates	112
Exercises	113
5.4 Reading and Writing Files	113
A. The see and seen Predicates	113
B. The tell and told Predicates	114
C. The seeing and telling Predicates	114
Exercises	115
5.5 Reading Programs	115
A. The consult Predicate	116
B. The reconsult Predicate	116
Exercises	118
5.6 Chapter Summary	119
A Look Ahead	119
References	120
 Chapter 6. Built-In Tools	
6.1 Built-In and User-Defined Operators	121
A. Types of Operators	121
B. Writing User-Defined Operators	122
Exercises	125
6.2 Equality	126
6.3 Program Control Tools	127
A. The Comma and Semicolon Operators	127
B. The cut Predicate	128
C. The fail Predicate	128
D. The true Predicate	129
E. The call Predicate	130
F. The not Predicate	130
G. The repeat Predicate	132
Exercises	133
6.4 Retaining Multiple Solutions	133

A. The findall Predicate	134
B. The bagof Predicate	134
C. The setof Predicate	135
Exercises	135
6.5 Chapter Summary	136
A Look Ahead	137
References	137

Chapter 7. More Built-In Tools

7.1 Assessing Prolog Terms	138
A. The var (Variable) and nonvar (Non-variable) Predicates	138
B. The atom, integer, float, number, and atomic Predicates	139
Exercises	141
7.2 Constructing and Decomposing Atoms: The Name Predicate	142
7.3 Accessing and Altering Clauses	143
A. The assert and retract Predicates	143
B. The listing Predicate	148
C. The clause Predicate	148
Exercises	150
7.4 Constructing and Decomposing Structured Objects	150
A. The =.. Predicate	150
B. The functor Predicate	153
C. The arg Predicate	154
Exercises	155
7.5 Debugging Tools in Prolog	156
A. The trace and notrace Predicates	156
B. The spy and nospy Predicates	158
Exercises	160
7.6 Chapter Summary	160
A Look Ahead	161
References	161

Chapter 8. Programming Techniques in Prolog

8.1 Principles of Good Programming	162
8.2 Program Layout	163
A. Comments	163
B. Meaningful Terms	164
C. Consistent Layout	164
D. Short Clauses	165
E. Limited Use of the Semicolon Operator	166
F. Careful Use of assert, retract, and cut	166
G. Utility Predicates	167
8.3 Programming-Efficiency Techniques	167
A. Procedural Program Avoidance	167
B. More Variables	168
C. Base and Recursive Cases	168
D. Left, Center and Tail Recursions	170
1. Infinite Loops with Left Recursion	170
2. Computer Memory Consumption with Recursion	171
3. Search Space Traversed with Recursion	171
E. Sub-goal Ordering Within a Clause	175
F. Iteration and Recursion	176
1. Iteration through the repeat-fail Combination	176
2. Iteration through assert Followed by the retract-fail	176

G. Drivers	177
8.4 Program Development	178
A. Evolutionary Approach	178
B. Modular Development Using Drivers	179
C. Problem Characterization	179
Exercises	179
8.5 Chapter Summary	180
A Look Ahead	181
References	181

Chapter 9. Prolog and LISP

9.1 LISP	182
A. Origins of LISP	182
B. The Language	183
1. LISP Data Types	183
2. Introduction to Built-in Procedures	184
3. Built-in Procedures in LISP and Prolog	185
4. Recursive Processing	187
5. Program Execution and Control in Prolog and LISP	188
6. More Complex Data Representations	189
9.2 Comparing Prolog and LISP	191
A. Understanding Language Development	191
B. Advantages and Limitations of Prolog and LISP	192
1. Simplicity and Ease of Use	192
2. Efficient Hardware Utilization	192
3. Language Flexibility and Power	192
4. Conclusions	193
9.3 Chapter Summary	193
A Look Ahead	193
References	194

PART TWO. INTRODUCTION TO AI PRINCIPLES

Chapter 10. Introduction to Artificial Intelligence

10.1 Introduction to Artificial Intelligence (AI)	197
A. Description of AI	197
B. A Brief History of AI	198
C. Uses of AI	202
D. Challenges in AI	202
Exercises	203
10.2 Introduction to Expert Systems	203
A. Description of Expert Systems	203
B. Uses of Expert Systems in Science and Engineering	204
C. Representing Knowledge in Expert Systems	206
D. Logic-Based Systems	207
1. Rule-Based Systems	207
2. Fuzzy-Logic Systems	208
E. Semantic Networks	210
F. Frame-Based Systems	213
G. Object-Oriented Programming	218
H. Blackboard Systems	226
Exercises	229
10.3 Chapter Summary	229

A Look Ahead	230
References	230

Chapter 11. Prolog in Artificial Intelligence

11.1 Search Strategies in Prolog	232
A. General Definition and Description of Search Strategies	232
B. Depth-First Search	233
C. Breadth-First Search	238
D. Best-First Search	240
E. Summary	241
Exercises	242
11.2 Knowledge Representation in Prolog	242
A. Rule-Based Knowledge	242
B. Semantic-Network Knowledge	246
C. Frame-Based Knowledge	247
D. Object-Oriented Programming	257
Exercises	259
11.3 Other Problem-Solving Strategies	260
A. AND-OR Strategies	260
B. Constraint Satisfaction	264
C. Generate-and-Test	266
1. Classic Generate-and-Test	266
2. Plan-Generate-Test	266
3. Hill-Climbing	267
D. Means-End Analysis	268
Exercises	270
11.4 Chapter Summary	271
A Look Ahead	272
References	272

PART THREE. AN ILLUSTRATED CASE STUDY: EXSEP— AN EXPERT SYSTEM FOR SEPARATION SYNTHESIS

Chapter 12. Introduction to EXSEP

12.1 The Application: Separation-Flowsheet Development and Multicomponent Separation Sequencing	277
A. Separation-System Synthesis or Separation-Flowsheet Development	277
B. Multicomponent Separation-Sequencing Problem	278
C. The Need for Efficient Problem-Solving Techniques for Separation Sequencing	279
D. Techniques for Separation Sequencing	279
E. Separation Heuristics and EXSEP	280
12.2 Introduction to Expert-System Development	281
A. Common Development Techniques	281
B. EXSEP's Development Plan	282
C. Limitations of Expert Systems	282
12.3 Chapter Summary	285
A Look Ahead	285
References	285

Chapter 13. Chemical Engineering Perspective of EXSEP

13.1 Introduction	287
A. Example 1: A Four-Component Separation	288
B. Challenges in the Synthesis of Process Flowsheets	292