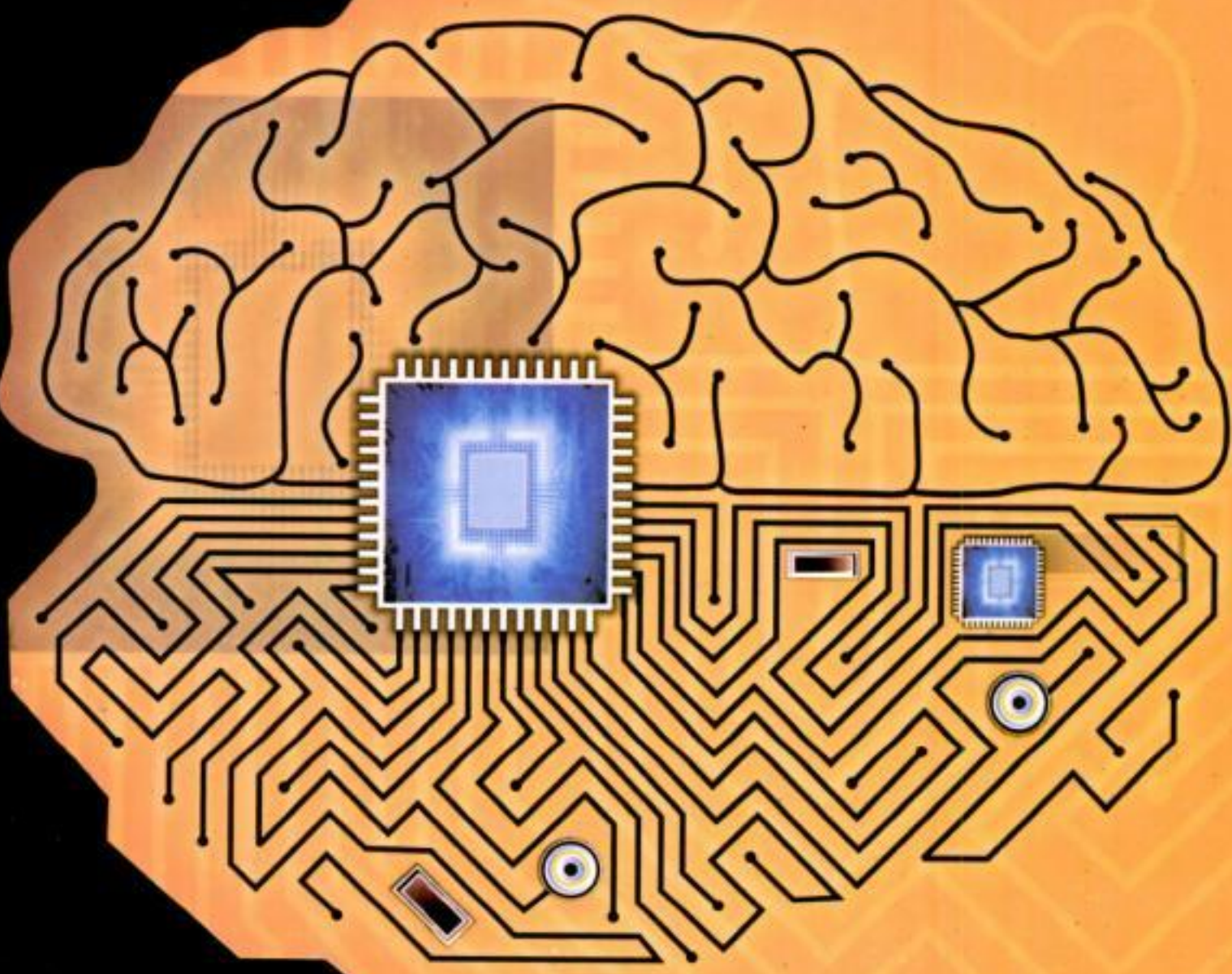
Eastern Economy Edition

Artificial Intelligence



R.B. Mishra

PHI Learning

₹ 325.00

ARTIFICIAL INTELLIGENCE
Ravi Bhushan Mishra

© 2011 by PHI Learning Private Limited, New Delhi. All rights reserved. No part of this book may be reproduced in any form, by mimeograph or any other means, without permission in writing from the publisher.

ISBN-978-81-203-3849-4

The export rights of this book are vested solely with the publisher.

Published by Asoke K. Ghosh, PHI Learning Private Limited, M-97, Connaught Circus, New Delhi-110001 and Printed by V.K. Batra at Pearl Offset Press Private Limited, New Delhi-110015.

Contents

Preface *xix*
Acknowledgements *xxiii*

Unit I: Logic Search and AI Languages

1. INTRODUCTION **3–25**

Intelligence *3*
Artificial Intelligence (AI) *3*
Components of AI *4*
The First Dimension (Core) *5*
The Second Dimension *6*
The Third Dimension *6*
History of Artificial Intelligence *7*
Salient Points in the Development of AI *10*
Knowledge and Knowledge-Based Systems *11*
Knowledge *11*
Knowledge-Based Systems *12*
AI in the Future *12*
Interface Design of CS *13*
Applications of AI *14*
Civil Engineering *14*
Computer Engineering *14*
Industrial Automation and Manufacturing *15*
Equipment Maintenance *15*
Management and Finance *15*
Office Automation *16*
Robotics *16*
Medical Computing and Informatics *17*
E-services *17*
Transportation *17*
Agriculture *18*
Oil Exploration, Minerals and Metallurgy *18*
Electrical Engineering *18*

Organization of the Book 19
Education and Research in AI 22
Pattern Recognition and Vision 22
Financial Applications 23
University of Edinburgh [AQR: 2003] 23
Centres in India 23
Other Centres 23
IBM 24
Summary 25
Exercises 25

2. LOGIC AND COMPUTATION **26–55**

Classical Concepts 26
Historical Development of Logic and Computation 29
Computational Logic 30
First-Order Logic (FOL) 33
Syntax and Semantics of FOL 33
Symbol Tableau 36
Herbrand Universe 40
Resolution 41
Resolution Strategy 42
Unification 43
Unification: Theoretical Concepts 44
Predicate Calculus in Problem-Solving 46
Model Logic 48
Semantics of Logic and Model Axioms 48
Godel on Provability as a Modality 50
Temporal Logic 50
Properties on Temporal Domain Structure 51
Types of Temporal Logic 52
Summary 53
Exercises 54

3. HEURISTIC SEARCH **56–75**

Search-Based Problems 57
Slide Bar Puzzle Problem (8, 15 and 24) 57
Tower of Hanoi 57
Monkey Banana Problem 59
Informed Search 60
Hill Climbing 60
Breadth First Search (BFS) 60
Depth First Search (DFS) 62
Evaluating Functions and Ordered Research 64

Informed Search 65
Best First 66
A* Search Algorithm 66
A* Algorithm 67
Water Jug Problem Using Breadth Search 71
Travelling Salesman Problem (TSP) 72
Branch-and-Bound Method 73
TSP Algorithm 73
Summary 74
Exercises 74

4. SEARCH IN GAME PLAYING **76–88**

AND/OR Graph 76
Tic-Tac-Toe 77
Minimax Problem 78
Alpha-Beta Search 79
Iterative Deepening 80
Puzzle Solving 81
Algorithm 81
MPC (Multi-Prob Cut) 83
Issues of RoboCup 85
AI versus Control Robots 86
Multi Robot Soccer 86
Summary 87
Exercises 88

5. ARTIFICIAL INTELLIGENCE LANGUAGES **89–120**

Functional Programing and LISP 89
LISP: Variable Assignment and Definition of Function 91
Input and Output Variables 96
PROLOG: Features, Syntax and Semantics 98
Salient Features of PROLOG 99
Structure of PROLOG 99
Programing in PROLOG 102
Object-Oriented Model (OOM) 109
Properties and Features of OOM 111
OOM and C++ 115
OOM: Program Implementation 118
Smalltalk 118
LISP Object System 118
Eiffel 118
Summary 119
Exercises 119

Unit II: Knowledge-Based Systems

6. KNOWLEDGE REPRESENTATION 123–147

Structure of an RBS *123*
Merit, Demerit, and Applicability of RBS *127*
Merits *127*
Demerits *128*
Types of Rules *129*
Applicability *129*
Semantic Nets *130*
Partitioned Semantic Nets *131*
Frames *134*
Conceptual Graphs *136*
CG to FOPL *137*
Restriction of Two Graphs *138*
Conceptual Dependency *139*
Scripts *144*
Reasoning with Script *145*
Summary *146*
Exercises *147*

7. AUTOMATED REASONING 148–176

Default Logic *148*
Problem for Default Reasoning *151*
Closed World Assumption (CWA) *151*
Predicate Completion *152*
Circumscription *152*
Default Reasoning *153*
Model-Based Reasoning (MBR) *156*
Modes of Diagnosis *159*
Case-Based Reasoning *159*
Reasoning Models *161*
Case Indexing *165*
Case Storage *166*
Case Retrieval *167*
Adaptation *169*
Multimodal Reasoning *171*
Combination of Rule-Based Reasoning and Case-Based Reasoning *171*
Abduction *172*
Truth Maintenance System (TMS) *173*
Summary *175*
Exercises *176*

8. PROBABILISTIC REASONING 177–192

Bayes' Theorem *177*
Bayesian Network *179*
Dempster and Shafer (DS) Theory of Evidence *182*
Combination of Belief Functions *182*
Upper and Lower Probability *184*
Comparative View of Uncertainty Measurement Methods *184*
Confidence Factor *186*
Deterministic Method *186*
Other Properties *188*
Probabilistic Logic (PL) *190*
Summary *191*
Exercises *192*

9. KNOWLEDGE ACQUISITION: MACHINE LEARNING 193–213

Introduction *193*
Knowledge Acquistion: A General View *194*
Knowledge Acquisition Process *195*
Automatic Knowledge Acquisition *196*
Function and Automatic Knowledge Acquisition *196*
Machine Learning *197*
Version Space *197*
Induction: ID3 *200*
Analogical Reasoning *202*
Explanation-Based Learning (EBL) *204*
Inductive Learning *205*
Knowledge Acquisition Tools *207*
Summary *212*
Exercises *212*

10. MULTI-AGENT SYSTEMS 214–245

Definition and Concepts *214*
AI Components in Multi-Agent Systems *218*
AOP: Components and Modalites *219*
Black Board Architecture *220*
Logical Modalities in MAS *223*
Agent Communication Languages *223*
ACL Platforms *230*
Theory of Cooperation *235*
Contract Net *236*
Game Theory *237*
Auction *238*
Computational Mechanism Design *240*

Conflict Resolution *241*
Application Models *242*
Intelligent Tutoring System (ITS) *242*
MAS for Semantic Web Service Composition *243*
Summary *245*
Exercises *245*

11. USER INTERFACE **246–265**

Introduction *246*
Computer Interface Specifications *247*
Human Factors *247*
Interface Design *249*
Developer Interface *249*
User's Interface *249*
System Interface *250*
Computing Methods for UI Design *251*
Vision-Based Interface *258*
Cognitive Models (CM) *258*
Vision: Jess GUI *260*
Natural Language *263*
Explanation *264*
Summary *265*
Exercises *265*

12. KNOWLEDGE-BASED SYSTEMS **266–288**

Introduction *266*
Structure of an Expert System *266*
Knowledge Base *266*
Knowledge Acquisition *267*
Expert Systems in Different Areas *267*
Expert System Shells *270*
Large Size *270*
Middle Size *275*
Small Size *276*
Comparison of Expert Systems *277*
Comparative View: Expert System Shell *279*
Ingredients of Knowledge-Based Systems *281*
Web-Based Expert Systems *282*
Overview: Web-Based Expert Systems *283*
Summary *286*
Exercises *287*

13. KNOWLEDGE DISCOVERY: DATA AND WEB MINING **289–304**

Introduction *289*
KDD Methods *290*
Data Mining Prediction Model *292*

Data Mining: Models and Algorithms 292
Knowledge Discovery by Association 296
Rule Generation and Induction 298
Rule Induction 299
AdaBoost 300
AdaCost 301
Data Mining Tools and Their Application 301
Web Mining 303
Summary 304
Exercises 304

14. WEB TECHNOLOGY, SEMANTIC WEB AND KNOWLEDGE MANAGEMENT **305–321**

Knowledge Representation: Ontology, Reasoning 305
Semantic Web Services Framework and Composition 307
Agents in Semantic Web 307
Advertisement Protocols Used by SP 308
Orchestration and Choreograph 310
Monitoring 310
Quality of Services (QoS) 311
Agent-Based Service Selection 311
Semantic Web Languages 312
Knowledge Management 315
Knowledge Management Task 315
Knowledge Management Interdisciplinary Structure 316
Components in KM 317
AI-Based Components 317
Knowledge Management and Web Technology: A Framework 319
Tasks and Tools 320
Intelligent Agents 320
Intelligent Intermediaries 320
Summary 321
Exercises 321

15. NATURAL LANGUAGE PROCESSING **322–362**

Computational Model of Language 322
Syntactic Structure and Analysis 323
Parsing 324
Case: Grammar 325
Grammar Types 326
Transition Network 327
Frame Representation 327
Augmented Transition Network (ATN) 328

Natural Language Processing Problems *329*
Document Summarization *329*
Information Categorization *331*
Information Retrieval *331*
Discourse Analysis *334*
Information Extraction *336*
Knowledge-Based System (KBS) Approaches in Natural Language Processing *337*
Machine Translation (MT) *338*
Direct Machine Translation *338*
Corpus-Based Machine Translation *339*
Learning Probabilities for Machine Translation *341*
Example-Based Machine Translation (EBMT) *341*
Matching Phase of EBMT *342*
Adaptation Phase of EBMT *344*
Recombination Phase of EBMT *345*
Knowledge-Based Machine Translation (KBMT) *346*
Rule-Based Machine Translation *347*
Transfer-Based-Machine Translation *349*
Divergence *349*
Structural Divergence *350*
Categorial Divergence *350*
Nominal Divergence *351*
Pronominal Divergence *351*
Demotional Divergence *352*
Conflational Divergence *352*
Possessional Divergence *353*
Performance Evaluation *353*
Blue 354
Unigram Precision and Recall Methods of Machine Translation Evaluation *355*
F-Measure and METEOR Score Methods *356*
Intelligent Computing Model from English to Sanskrit Machine Translation *356*
ANN-Based Modelling *359*
Summary *361*
Exercises *361*

16. DEVELOPMENT, SELECTION AND EVALUATION **363–383**

Development Phases *363*
Problem Description *363*
Problem Selection *364*
Expert and Domain Characteristics *365*
Cost/Benefit *365*

Knowledge-Based System Level Methods 366
Trillium 366
MK Model 366
Miller Model 368
Task-Based Specification Methodology (TBSM) 368
Decision Table-Based Design 369
Component Selection 372
Rules for Inference 373
Rules for Search 373
Rules for Interface 373
Rules for User's Interface 374
Validation 374
Quantitative Methods 375
Verification 376
Rule-Based Model 376
Heuristic Model: Performance Model 378
Performance Evaluation 379
Evaluation and Validity Assessment 382
Summary 383
Exercises 383

Unit III: Artificial Intelligence and Knowledge-Based Systems Applications

17. SOFTWARE ENGINEERING AND ARTIFICIAL INTELLIGENCE 387–406

Introduction 387
Comparative View: SE and AI (ES) 387
KBS Approaches in Software Engineering (SE) 388
SE: Requirement Analysis—Specification 389
SE: Design 392
Reusability 393
Reusability and Knowledge-Based System 395
AI and Object-Oriented Design 397
Design Patterns 399
KBS Metrics for Software Agents 400
Necessity of AI to Software Engineering 401
What AI Can Accomplish 402
Software Engineering and Agent-Based System 402
Summary 405
Exercises 405

18. ARTIFICIAL INTELLIGENCE IN MEDICINE **407–425**

Introduction *407*
RBS in EMG/ECG/EEG *408*
EMG *408*
ECG: Parameter Detection and Interpretation *410*
EEG *411*
CASNET, INTERNIST and PIP *414*
CASNET *414*
INTERNIST *414*
Present Illness Program (PIP) *414*
RBS Model in MYCIN *415*
Health Information System and the Hospital Information Systems *417*
Frame-Based Patient Care Model *419*
KBS, CBR and Multimodal Reasoning in Medicine *420*
Case-Based Reasoning (CBR) *420*
Summary *424*
Exercises *424*

19. INDUSTRIAL AUTOMATION: FMS AND ROBOTICS **426–448**

Introduction *426*
Process Planning *427*
Problem Description *427*
Design Strategies *427*
Rule-Based Model *427*
Selection of Part-Type *428*
Selection of Machine Tools *428*
Structure of FMS *428*
AI-Based Manufacturing: A Comparative View *430*
Process Planning *431*
Process Scheduling *432*
Knowledge Base *433*
Knowledge-Based Scheduling System *434*
Industrial Process Monitoring Control: An Example *436*
Generation of Rules *437*
Monitoring Equipment *438*
Object-Oriented Model (OOM) for Process Monitoring *439*
Robotics, Artificial Intelligence, and Industrial Automation *442*
Planning, Assembly and Control *443*
Planning and Assembly *443*
Knowledge Representation *444*
Kinematics and Control *444*
Summary *447*
Exercises *447*

20. ELECTRONICS COMMUNICATION **449–465**

Electronics: OOM and RBS *449*
ES for VLSI Reverse Engineering *451*
Case-Based Reasoning and PCB *452*
Network Designing *453*
Expert System Component *454*
Network Management *455*
Performance Management *455*
Configuration Management *456*
Accounting Management *456*
Fault Management *456*
Security Management *456*
Overview of Fault Management Activities *457*
Intelligent Fault Management Systems *457*
Rule-Based Techniques in Automated Network Management *458*
Salient Features *458*
System Design and Implementation *459*
Problem Descriptions *459*
Simulation Environment *459*
Knowledge Base *460*
Problem Inference Engine and Problem Viewer *460*
Electrical Systems *461*
ES for Power Systems (PS) *461*
Power System Planning *461*
NOMOS: Fault Diagnosis *461*
CBR: Power Restoration *462*
User Interface *462*
RBS: Power Plant *463*
Summary *465*
Exercises *465*

21. MANAGEMENT AND BUSINESS INTELLIGENCE **466–485**

AI-Based Decision Analysis *466*
Decision Support System (DSS) *467*
Components of DSS *468*
AI-Based DSS: Models and Examples *469*
CBR and DSS *470*
Knowledge Management *471*
Financial Applications *472*
Contax: An Income Tax Planning Consultant *472*
FOL in IFPS *474*
AI Methodology in Customization *474*
Integration of E-Commerce, Web Technology and KBS *475*
KBS in CRM *476*

Architectural Components of a CRM Solution *477*
Application of ICS: ANN *479*
Knowledge-Based System (KBS) *479*
User's Interface *480*
Outward View: Web Infrastructure and Technology *480*
Defining the Technology Architecture *482*
Query, Analysis, Segmentation *482*
Features of CRM *482*
Personalization *482*
Fast Resolution *482*
Analytical Approach *483*
Information Arena *483*
Technology Architect *483*
Spectrum of Intelligent Business *483*
Summary *484*
Exercises *485*

REFERENCES **487–500**

INDEX **501–503**

Preface

The coining of the term *Artificial Intelligence* originated in an IBM-sponsored summer workshop at Dartmouth, New Hampshire, USA, in June 1956. Among the participants in the seminal conference were Claude Shannon from IBM, John McCarthy from MIT, and Edward Feigenbaum from Stanford University, along with Herbert Simon and Allen Newell, and many others. Artificial intelligence combines and encompasses many concepts and methods deployed by researchers in many diverse fields of computation and cognition. Since its origin, many evolutionary conceptual changes in its theory and practice have been incorporated from time to time. It is difficult to present all the conceptual changes and practices in this area in a single book of this nature. However, an attempt has been made to present as many theoretical concepts and their applications as possible for the understanding of undergraduate and postgraduate students, and researchers in the discipline of computer science and engineering, and artificial intelligence and its applications to various fields.

In December 1984, I had an opportunity to attend a one-day workshop on "Experts Systems" organized by the Indian Institute of Science Bangalore. The lecture was delivered by Prof. H.N. Mahabala, the then Professor in IIT Madras, and another expert from overseas with a bearded face (perhaps Hayes Roth, with due apologies if I have mistaken his name). I do not recall what I understood at that time but definitely would like to present the concluding comments of Prof. Mahabala:

"We have conducted this workshop like Laurel and Hardy, the famous pair of comedy of the 1940s and 1960s."

Then onwards I nursed a desire to work in this area. This book is one of the products of my three decades of dedication to teaching and research in the area of artificial intelligence.

The book has been written keeping in mind the interest and inclination of students and naïve researchers studying the courses in artificial intelligence and related subjects from the disciplines of computer science and engineering, industrial engineering, electrical engineering, biomedical engineering, electronics and communication engineering.

The book has been organized into three basic units: Logic and Search, Knowledge Based Systems, and Artificial Intelligence and Knowledge Based Systems. In general, the core concepts of AI are logical computation and knowledge. Other areas are natural language processing, vision, speech recognition, robotics, and pattern recognition.

Unit I contains five chapters. Chapter 1 covers the basic definitions and illustrations of intelligence, artificial intelligence, knowledge, knowledge based system (KBS), the origin of AI

and the chronological development in the concepts of core and peripheral components that constitute and contribute to AI formalism up to date. The correlation between AI and KBS, and the outline of domain of AI and KBS in the industry and research organizations have been described. Chapter 2 deals with computational logic and reasoning in detail. Chapter 3 contains search methods and related problems. Some heuristic methods based on believed search, i.e. breadth first, depth first, best first and hill climbing are discussed with some of the problems of AI origin such as puzzles and travelling salesman. The problem reduction methods and some game playing problems are described in Chapter 4. The min-max problem, alpha beta cut, the iterative deepening A*, the state-of-the-art in game playing and their solution strategy are illustrated with enumeration of their computational complexities. Chapter 5 covers functional and logical programming with their semantics, structures and constructs. An attempt has been made to deal with the concepts and programming techniques of LISP and PROLOG. Another aspect which does not purely belong to AI community, but has come close to AI programming methodology in due course of time, the object oriented model (OOM) and programming has also been covered.

Unit II covers most of the prevailing modeling and computational concepts of knowledge such as representation, reasoning, acquisition, evaluation, management discovery. Formal and informal methods of knowledge representation such as predicate calculus, rule based system (RBS), semantics nets (SN), frame, conceptual graph (CG), conceptual dependency (CD) and scripts have been discussed. The correlation among these is mentioned in Chapter 6. Chapter 7 covers monotonic and non-monotonic reasoning based on default logic, closed word assumption predicate computation circumscription. The model-based and case-based reasoning, truth maintenance system are also discussed with examples in this chapter. The uncertainty measurement in imprecise and incomplete knowledge is performed by Bayes' probabilistic method, certainty factor and theory of evidence given by Dempster and developed by Safer in due course of time. These aspects of uncertainty handling are described in Chapter 8. Knowledge acquisition methods, machine learning, multi-agent systems and distributed AI have been dealt in Chapters 9 and 10. Different methods of users interface design, single and multimode modalities have been described in Chapter 11. In Chapter 12, expert shells for the design of KBS have been developed from the mid-1960s to up to date in the various fields of engineering, science, medicine; hardly any field of human endeavour is left untouched by the realm of knowledge engineering. Chapter 13 deals with the knowledge discovery and KDD data mining. These two concepts are inseparable, in true sense of principle and practice as well. Clustering graph based and association rules and other heuristic methods of knowledge discovery and data mining have been covered in this chapter. Knowledge management is a forum for research and education with applications in information management, industries and organization of finance and business; these issues are described in Chapter 14. One of the used orbital processing elements for AI paradigm is the written language, i.e. text. Language translation understanding, information retrieval, document summarization, and discourse analysis are vital aspects of natural language processing. Conventional methods, particularly syntactic and symmetric analyses with their structure and grammar, are described in Chapter 15. An attempt has been made to cover the emerging concepts and methods, up to date, for discourse analysis, information extraction and summarization of documents with KBS methodology. Statistical rule based and exemplar based machine translation methods are dealt with in this chapter.

The design steps of KBS and the selection of a particular ES shell and the KBS components for a particular problem domain are described in Chapter 16. The various phases in the development of KBS, together with their comparative view to software engineering, have also been discussed. The development phases such as requirement analysis, selection of experts, and selection of KBS components for prototype design, implementation of design, its formalization, performance evaluation and maintenance strategy have been dealt in detail.

Unit III (Chapters 17–21) is concerned with the description of application domains of AI in various disciplines of software engineering, medicine, industrial automation (robotics and FMS) networking, electrical power system, etc.

The book has been written in a style comprehensible to senior undergraduate students, postgraduate students and researchers, who want to study AI and its applications in various disciplines.

Thanks are due to PHI Learning for accepting to publish this book.

Ravi Bhushan Mishra

Acknowledgements

At the outset, I would like to thank the authors of two books which helped me in learning the vital concepts of the intriguing subject like Artificial Intelligence. The first book *Expert System Design & Application* by Weiss and Kulikosky, was given to me in 1986 by Radhakrishnan (forgive me for not being able to recall his initials), an erstwhile PhD scholar (in the Department of Mechanical Engineering and Department of Computer Engineering) at IIT Delhi. The second book, *MYCIN* by E. Shortliffe was given to me by Professor A.K. Mahanta (presently at IIT Guwahati). These books really increased my knowledge of the subject of AI. I also owe a special gratitude to both of my friends who lended the above books to me.

I also received valuable assistance from my undergraduate and postgraduate students, who did their dissertation work under my supervision during 1988–1992, in the area of AI-based computing in medicine. Particularly, among them, I wish to mention Prof. S. Radhakrishnan, Computer Engineering Department, Kalasalingam University, Tamil Nadu, Prof. Samarendra Dandapat, Electronics and Communication Engineering, IIT Guwahati, P.N. Shah (now in the USA), A.A. Khalili (from Iran), R.S. Rao (now in the USA), Subramania Anoop, Biomedical Engineering Section, SGPGI, Lucknow. I extend my utmost thanks to all of them.

Some of my PhD scholars who did their work under my supervision from 2006 to 2010, including Sandeep Kumar, Babita Pandey, Bireshwar Dass Mazumadar, Vimal Mishra, Kiran Mishra deserve credit for their contribution to the preparation of the book.

Professor Anil Kumar Tripathi needs a special mention. When I shared with him that I wanted to write a book on AI, he promptly responded *Likh Daaliye* (write down). His style of encouragement boosted my confidence to take up this project.

I would be failing in my duty if I did not acknowledge the support I received from my family: my wife, Sushila, and our two sons. I also wish to record my gratitude to my Bhabhiji, Jayanti, and elder brother, Shashi Bhushan, both of them showered love on me when I was a student.

The present PhD batch of my students, especially Md. Sami (from Yamen), Akash Dubey and Shahnawaz Khan (from India) deserve thanks for the help they have rendered in the preparation of the manuscript.

I extend my heartfelt thanks to Shri Rajendra Prasad Khare (STA in our department), who has been my bosom friend since 1978 and Mrs. Usha Yadav, a laboratory staff.

Finally, I thank the staff of PHI Learning, particularly the Editorial and Production Departments for giving the present shape to the book.



UNIT I

Logic Search and AI Languages

Copyrighted material

Chapter 1



Introduction

INTELLIGENCE

Intelligence is a quick and accurate response to a stimulus. As in the examination, scoring of marks up to a certain level of excellence measures the intelligence of the person concerned. The requisite score has to be obtained in a given time. How accurately one answers a question is important. Stipulated or scheduled time corresponds to quickness, and the right answer to accuracy. This is what we observe in practice, day-to-day activities too. There are two other domains of knowledge, i.e. philosophy and psychology, which conceptualize and define the term *Intelligence* in a different way; but it turns out to be the same manifestation (understanding) as it is accepted by the common man. Intelligence is concerned with *reasoning* in general and logical approach in particular. Even if one answers accurately a numerical problem but due to lack of logical approach to the problem, he or she may not score equal or more in comparison to the person who makes a logical approach but with a wrong answer. Logical approach with approximate numeric answer in philosophy and, also in practice, is associated with the level of appreciation of the actor (who works).

In psychology, intelligence involves learning, adaptation and self-organization in response to known or unknown situations or stimuli. The response to a learnt situation, through adaptation and self-organization to an unknown situation, displays the sign of intelligence for humans as well as animals.

These concepts in philosophy and psychology correspond to human beings, but not to machine, the inanimate entity, which is still an open problem.

ARTIFICIAL INTELLIGENCE (AI)

The art and science of bringing learning, adaptation and self-organization to the machine is the act of *artificial intelligence*. Artificial intelligence, in essence, is not only the science of computation but also the logic of cognition. As the information processing paradigm of human beings induced the model of a computing system, likewise the logic of cognition generated the methodologies for the origin of AI and its dimensions.

Copyrighted material

The IBM sponsored a summer workshop at Dartmouth, New Hampshire, USA, in June 1956. Pioneer researchers in the area of computer science, logic, geometry, mathematics and cognitive psychology gathered to discuss the specific topics such as automatic theorem proving and new programming languages. Among the participants in the seminal conference were Claude Shannon from IBM, John McCarthy from MIT, and Edward Feigenbaum from Stanford University, along with Herbert Simon and Allen Newell, and many others. The important outcome of the conference was the coining of the term *Artificial intelligence*, which encompasses many concepts and methods deployed by researchers in many diverse fields of computation and cognition.

COMPONENTS OF AI

The core components and constituents of AI are derived from the concept of logic, cognition and computation; and the compound components, built-up through core components are knowledge, reasoning, search, natural language processing, vision, etc. as shown in Table 1.1.

Table 1.1 Three-level components of AI

Level	*Core*	*Compound*	*Coarse components*
Logic	Induction Proposition Tautology Model logic	Knowledge Reasoning Control Search	Knowledge-based systems Heuristic search Theorem proving
Cognition	Temporal Learning Adaptation Self-organization	Belief Desire Intention	Multi-agent system Cooperation Coordination AI programming languages
Functional	Memory Perception	Utterance	Vision Natural language Speech processing

The core entities are inseparable constituents of AI in that these concepts are fused at atomic level. The concepts derived from logic are propositional logic, tautology, predicate calculus, model and temporal logic. The concepts of cognitive science are of two types: one is functional which includes learning, adaptation and self-organization, and the other is memory and perception which are physical entities. The physical entities generate some functions to make the compound components.

The compound components are made of some combination of the logic and cognition stream. These are knowledge, reasoning and control generated from constituents of logic such as predicate calculus, induction and tautology and some from cognition (such as learning and adaptation). Similarly, belief, desire and intention are models of mental states that are predominantly based on cognitive components but less on logic. Vision, utterance (vocal) and expression (written) are combined effect of memory and perceiving organs or body sensors such as ear, eyes and vocal. The gross level contains the constituents at the third level which are knowledge-based systems (KBS), heuristic search, automatic theorem proving, multi-agent systems, AI languages such as PROLOG and LISP, Natural language processing (NLP). Speech processing and vision are based mainly on the principle of pattern recognition.

AI Dimension: The philosophy of AI in three-dimensional representations consists in logic, cognition and computation in the x-direction, knowledge, reasoning and interface in the y-direction. The x–y plane is the foundation of AI. The z-direction consists of correlated systems of physical origin such as language, vision and perception as shown in Figure 1.1.

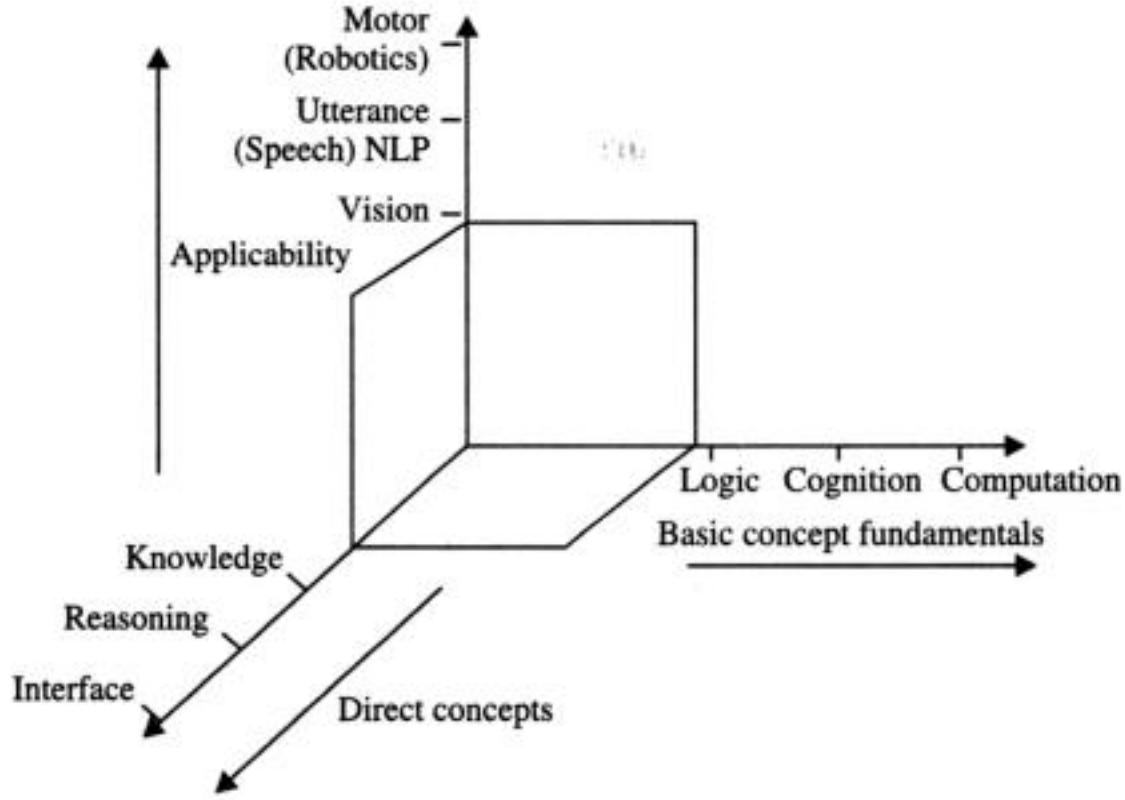


Figure 1.1 Three-dimensional model of AI.

The First Dimension (Core)

The theory of logic, cognition and computation constitutes the fusion factors for the formation of one of the foundations on coordinate x-axis. Philosophy from its very inception of origin covered all the facts, directions and dimensions of human thinking output. Aristotle's theory of syllogism, *Descartes* and Kant's critic of pure reasoning and contribution of many other philosophers made knowledge-based on logic. It were Charles Babbage and Boole who demonstrated the power of computation logic. Although the modern philosophers such as Bertrand Russell correlated logic with mathematics but it was Turing who developed the theory of computation for mechanization. In the 1960s, Marvin Minsky pushed the logical formalism to integrate reasoning with knowledge. The contribution of philosophers, mathematicians and psychologists to computational logic is discussed in Chapter 2.

Cognition

Computers became so popular in a short span of time due to the simple reason that they adapted and projected the information processing paradigm (IPP) of human beings: sensing organs as input, mechanical movement organs as output and the central nervous system (CNS) in brain as control and computing devices, short-term and long-term memory were not distinguished by computer scientists but, as a whole, it was in conjunction, termed *memory*.

In further deepening level, if one goes into the IPP, the interaction of stimuli with the stored information to produce new information requires the process of learning, adaptation and self-organization. These functionalities in the information processing at a certain level of abstraction of brain activities demonstrate a state of mind which exhibits certain specific behaviour to qualify as intelligence. Computational models were developed and incorporated in machines which mimicked the functionalities of human origin. The creation of such traits of human beings in the computing devices and processes originated the concept of intelligence in machine as virtual mechanism. These virtual machines were termed in due course of time *artificial intelligent machines*.

Computation

The theory of computation developed by Turing—finite state automation—was a turning point in mathematical model to logical computational. Chomsky's linguistic computational theory generated a model for syntactic analysis through a regular grammar.

The Second Dimension

The second dimension contains knowledge, reasoning and interface which are the components of knowledge-based system (KBS). Knowledge can be logical, it may be processed as information which is subject to further computation. This means that any item on the y-axis is correlated with any item on the x-axis to make the foundation of any item on the z-axis. Knowledge and reasoning are difficult to prioritize, which occurs first: whether knowledge is formed first and then reasoning is performed or as reasoning is present, knowledge is formed. Interface is a means of communication between one domain to another. Here, it connotes a different concept then the user's interface. The formation of a permeable membrane or transparent solid structure between two domains of different permittivity is termed *interface*. For example, in the industrial domain, the robot is an interface. A robot exhibits all traits of human intelligence in its course of action to perform mechanical work. In the KBS, the user's interface is an example of the interface between computing machine and the user. Similarly, a program is an interface between the machine and the user. The interface may be between human and human, i.e. experts in one domain to experts in another domain. Human-to-machine is program and machine-to-machine is hardware. These interfaces are in the context of computation and AI methodology.

The Third Dimension

The third dimension leads to the orbital or peripheral entities, which are built on the foundation of x–y plane and revolve around these for development. The entities include an information

system. NLP, for example, is formed on the basis of the linguistic computation theory of Chomsky and concepts of interface and knowledge on *y*-direction. Similarly, vision has its basis on some computational model such as clustering, pattern recognition computing models and image processing algorithms on the *x*-direction and knowledge of the domain on the *y*-direction.

The third dimension is basically the *application domain*. Here, if the entities are near the origin, more and more concepts are required from the *x*–*y* plane. For example, consider information and automation, these are far away from entities on *z*-direction, but contain some of the concepts of cognition and computation model respectively on *x*-direction and concepts of knowledge (data), reasoning and interface on the *y*-direction.

In general, any quantity in any dimension is correlated with some entities on the other dimension.

The implementation of the logical formalism was accelerated by the rapid growth in electronic technology, in general and multiprocessing parallelism in particular.

HISTORY OF ARTIFICIAL INTELLIGENCE

The development period from its origin to up-to-date has been divided into nine phase with an interval of five years: 1956–61, 1962–67, 1968–73, 1974–79, 1980–85, 1986–91, 1992–97, 1998–2003, 2004 and onwards.

1956–61: The first year of this period gave rise to the terminology 'Artificial Intelligence' proposed by McCarthy and supported by the participants in the conference. In the same year Samuel developed a program for chess playing which performed better than its creator.

Around 1956–57, Chomsky's grammar in NLP, i.e. linguistic model processing was a remarkable event. In 1958, McCarthy made a very significant contribution, development of LISP, an AI programming language and advice taker which combined the method of knowledge representation and reasoning. Herbert Gelerriter at IBM in 1959 designed the first written AI program for geometry theorem proving in quick succession of time. In 1960, Widrow alone and then with Hoff developed networks called 'Adaline', based on the concepts of Hebbian learning. In 1956–57, Logic theorist (LT), a program for automatic theorem proving was developed.

1962–67: At the beginning of this period Frank Rosen blatt proposed the concept of 'perception' in the line of Widrow's concept for artificial neural networks (ANN), a biological model to incorporate computational rationality. In 1963, McCarthy developed a general purpose logical reasoning method and it was enhanced by the Robinson's 'Resolution principle' (Robinson, 1965). The logical neural model of McCulloch and Pitts was enhanced by Winograd and Cowan in 1963. James Slage's program was developed for the interpretation of calculus in 1963. In 1965, Hearsay was developed at CMU for natural language interpretation of subset language.

1968–73: In this period, some AI programs for practical use were developed. In 1967, David Bobrow developed 'STUDENT' to solve algebra story problems. The first knowledge-based

expert system DENDRAL was developed by J. Lederber, Edward Feigenbaum and Carl Djerassi in 1968, although the work had started in 1965. The program discovered the molecular structure of an organic compound based on the mass spectral data. MACSYMA, another expert system (ES) which solved numerous types of mathematical problems was written in LISP, and was developed by Carl Englishman, William Martin, and Joel Morris at MIT. Early Perceiver and Memorizer (EPAM) was written in Information Processing Language (IPL) and was developed by Feigenbaum and Simon in 1970. Patrick Wilson gave rise to the machine learning theory. Vision project of David Huffman also came into the picture. In 1972, Terry Winograd made a significant contribution to natural language understanding.

The most important contribution to AI came from Newell and Simon in the form of General Problem Solver (GPS) written in IPL, a program which simulated some mental activities of human beings in solving complex problems. Feigenbaum's Heuristic Programming Project (HPP) at Stanford University in 1969 also drew the attention of researchers in AI. During 1972–73 some expert system was developed: AG eat for ES generation tool, Teiresias knowledge transformation tool and PROSPECTOR as mineral exploration were developed at Stanford University.

1974–79: In this period the most significant and widely acclaimed product in AI was MYCIN. In Stanford University in 1974–75, medical personnel Edward Shortliffe joined hands with computer scientist Buchanan to develop a rule-based model MYCIN, which was used for detection, diagnosis and treatment of anti-bacterial infection. Around the same period, Scott Falman designed 'PLANNER' which became popular in the USA. Marvin Minsky developed a knowledge representation model 'FRAME' in 1975 which treated objects as frames, its parts as slots and properties as attributes, arranged in a large hierarchical taxonomy.

In 1978, Crany Henderix developed Language Interface Facility with Ellipsis and Recursion (LIFER) to be used as interface to database management. Roger Schout emphasized the knowledge representation and reasoning rather than syntax in natural language processing. Reiger designed human memory organization model in 1976. Many expert system programs such as GPS 5 at CMU, AES building tool, Cadusues at the University of Pittsburgh for diagnosis of internal medicine and Rosie at Rand, an expert system building tool were developed in this period.

1980–85: XCON, also termed R1 was an expert system program designed and developed by McDermott of CMU and DEC's engineers in the early 1980s to generate a computer configuration for the given customer's requirement. In 1981, the fifth generation of computers, incorporating AI concepts and programming language Prolog was announced. In 1982, John Hopfield on the basis of statistical mechanics formulated the storage and optimization properties of neural network interconnection.

In this period many expert system shells, expert system tools and expert system programs were developed. During 1984–85 the expert system shells came into picture was EMYCIN by Buchanan, a rule-based diagnostic consultant based on LISP, EXPERT by Weiss and KAS by DUDE are the rule-based model for classification using FORTRAN; a semantic network based system using LISP, others are knowledge crafts by GILMORE using object-oriented programming (OOP), KL-ONE by Brakeman using LISP for automatic inheritance. The most important development was PROLOG as AI programming language by Clockcin 1984.

1986–91: In this period significant developments occurred in ANN model in particular, the appearance of error back propagation algorithm formulated by Rumelhart and Hinton in parallel distributed processing. The probabilistic reasoning method in intelligent system appeared in 1988 by the work of Pearl. The distributed artificial intelligence concepts were formally incorporated in the multi-agent systems. The complete agent-based architecture was first implemented in a model SOAR, designed by Newel, Laired and Rosenbloom. Hidden Markov Model (HMM) was also conceptualized for speech processing and natural language processing during this period.

1992–97: This period was in the full swing of rise to the agent-based technology and multi-agent system (MAS). In 1992, Nawana brought the concept of autonomous agent, capable of acting independently with rationality. Different kinds of agents were defined. In 1994, Jennings Yoam introduced social or responsible agents, Yoam Shoham in 1993, described the concept of agent-oriented programming with different components and modalities. Belief, desire and intention (BDI) theory was introduced by Cohen (1995) during this period. The concept of cooperation, coordination and conflict resolution in MAS was introduced in this period.

In the NLP, a mean X-project was developed at Zurich by Nobel laureate Gerd Binning, who emphasized the use of word 'knowledge' to achieve comprehension. Gordon made a model language for representing strategies on standard AI planning techniques. The ABLE (Agent Building and Learning Environment) was developed by Joe Bigns, which focused on building hybrid intelligent agents for both reasoning and learning.

1998–2003: The introduction, incorporation and integration of AI concepts, theories and algorithms in and with web technology for information retrieval, extraction and categorization, document summarization, machine translation (single or multilingual) discourse analysis were performed in this period.

Courteous logic program in which users specify the scope of potential conflict by pair-wise mutual exclusion is implemented in common rules, a Java library used for e-commerce, business and web intelligence emerged as the front forum of AI. Formula Augmented Network (FAN) was developed by Morgenstern and Singh, is a knowledge structure, which enabled efficient reasoning and about potentially conflicting business rules.

Heuristic search methods were devised for game playing such as chess, checkers, Rubik cube with the concept of MPC.

Blue Deep was a chess-playing computer developed by IBM on May 11, 1997, the machine won a six-game match by two wins to one with three draws against world champion Garry Kasparor. Robotics in game playing and surgery marks the splendid achievements in AI based robotics. The Seoul Robotic Football Game from 2001 onwards regularly updated is a good example of the development and incorporation of AI search methodology in game playing.

2004–Future directions and dimensions: In Communications of the ACM 2003, a certain direction and dimension of AI have been envisaged for the future. Shannon has given the frame qualification and ramification and learning from books, i.e. reading the text and extracting relevant information. Three-dimensional robot surrounding in distilling from the www, a huge knowledge base, the development of semantic scrapes and computational engines such as

human mind and brain. Knowledge discovery and vision system for biometric and automated object regulated in supermarkets are important milestones.

Intelligent interface design should be intuitive for the novice, efficient perception for expert and robust undermines which would facilitate recovery from cognitive and manipulative mistakes. Programs that are helpful for diagnosis of errors and suggestions for corrective actions are to be developed.

SALIENT POINTS IN THE DEVELOPMENT OF AI

Salient points in the development of AI depicting the contributors, their contributions and the years are shown in Table 1.2.

Table 1.2 Main developments in AI

Name	*Contribution*
Samuel (1952)	His program quickly learnt to play better than the creator itself
McCarthy (1956)	High level language LISP
McCarthy (1958)	A knowledge enrich and reasoning in formal logic
Gelernter et al.(1959)	Geometry theorem power
Slagle (1963)	A silent program for elementary integral calculus problem solution.
Evans (1968)	Analogy geometry problems for IQ tests
Bombrow (1967)	Student–algebra story problem
Winston (1970)	Learning theory
Winograd (1969)	Natural language understanding
Minisky and Papert (1969)	Fifth generation project. AI for chip design and a human–computer user interface in US
Hopfield (1982)	Developed Hopfield network for optimization using methods for statistical mechanics
Rumelhart et al. (1986)	ANN method for memory in distributed processing
Heckerman (1986)	Learning form experience
(1987)	Hidden Markov Model: Hypothesis must be subjected to rigorous empirical experiment and the result must be analyzed statistically for its importance
Pearl (1988)	Bayesian network for uncertain reasoning theory to AI
(1989)	Probabilistic reasoning in intelligent system ANN application to speech data mining
Laird et al. (1987)	Rational agent SOAR: Best known architecture of agent
Russell and Norvig (1995)	Agent perspective has drawn AI close to control theory, economics
Nilsson (1998)	Cross and logistic planning and scheduling DART: Dynamic analysis and replanning
Kaelbling et al. (1996)	Via robotic control hip replacement prosthesis crossword problem
Morgenstern and Singh (1999)	Formula Augmented Network for reasoning in business rules, knowledge discovery in grids
Campbell et al. (2002)	Hardware chip for game playing. Semantic nets., 3d robotics, robotic game playing robotic surgery
(2002)	Hepatic, semantic, web intelligence and web-based ES shells

KNOWLEDGE AND KNOWLEDGE-BASED SYSTEMS

Knowledge

In philosophy, abstract terms are given with their manifestations rather than definitions. Physical systems are well defined, functional entities are well illustrated but entities combined, or being constituted with functional attributes, abstraction and functionalism are mainly manifested and illustrated. Knowledge combines abstraction and functionalism. Thinking and learning are the abstraction of mental activities; speech, vision and movement are the functional activities whereas information and knowledge are the compound entities made of abstraction entities and functional activities.

Knowledge in philosophy

Knowledge in philosophy is addressed primarily by two eminent schools of thoughts: one says knowledge is what one experiences, whereas the other says what one reasons out is knowledge. It was German philosopher Immanuel Kant who combined both the concepts and proposed that knowledge is made of experience and reasoning, and what the sensing organs perceive is an experience. In the night, if one sees (perceives) a humanlike structure then they may say that it is a man or woman. If it neither moves nor produces any symptom or signal which emanates from human beings, then the person is compelled to think or reason out something else. They may conclude with some reasoning that it is a statue. Thus, declaration of the structure as statue is based on experience (perception) and reasoning (instantiation of thinking).

Knowledge in science

The scientific community has not paid much attention and devotion towards formulation of knowledge as they have paid to other entities. Medical science in general and neuroscience and neurology in particular have worked out the changes in the architectural feature of neurons. Neurons are the entities in brain which change their signal or voltage level of excitation and their interconnection according to the modalities of the stimuli as well as some internal processes, pertaining to thinking or cognitive activities. The excitation level of neurons encodes information and does the coding, i.e. encoding (writing) and decoding (reading) contribute to the knowledge. The research and education in neurophysiology which contribute to the understanding of encoding and decoding has generated the computation model of brain on the basis of artificial neural networks (ANN).

Knowledge in engineering

Engineering emphasizes the implementation of the scientific outcome through its principle of approximation. The ANN is a data prone model of computation, which manipulates data pertaining to knowledge in a particular domain. The development of a symbol is a knowledge prone computational model implemented on AI-based symbolic machine. Engineering has transformed the data-based computation to knowledge-based computation. In engineering, knowledge is not defined as in philosophy, but knowledge is coded into numeric terms and then numeric terms or knowledge information is transformed to knowledge through heuristics and

algorithms. Thus, engineering is concerned with transformation of data to knowledge and vice versa through mathematical, heuristics and algorithms steps either individually or in combination.

Knowledge-Based Systems

First, we illustrate the term *system* and then *knowledge-based system* (KBS) as knowledge has already been illustrated in different contexts earlier. *System* is an interconnection of different entities, atomic in nature to form as a whole in due course of time for a specific goal in mind. For example, a library is a system, which is the interconnection (not in physical) of books, journals, tapes, students, users and staff at different levels of abstraction. A car is a system having different parts or entities such as wheels, seat, engine, chassis, body and others. The atomicity or granularity of the concerned entities is arbitrary. An engine, for example, may be defined at atomic level in one system, but its different parts, e.g. piston, value and chamber may be defined as subsystem in other contexts when one takes engine as a system.

The interaction or interconnection of entities among themselves as abstract entities or physical entities or among the abstract and physical gentiles is termed *system.*

Who contains the knowledge of driving a car? The driver, if 'yes', then is it a knowledge-based system. Yes, in some sense. If a trained robot, then driving the car with robot is replacing a driver pertains to knowledge-based system. The knowledge of driving, for example, is being transferred to the robot through some mechanism.

Edward Feigenbaum at the International Conference on Artificial Intelligence in 1977 formalized in his research paper that the transferring expertise of a person to a computational model be implemented on computing machine. Although the terms, *expert system* and knowledge-based system are used interchangeably, these have the same manifestation in formalism and practice.

The basic objective of a knowledge-based system is to store facts and figures for a particular application domain in a memory, either statically or dynamically, and to derive new factors or conclusions when the knowledge of the domain is incomplete and imprecise knowledge is supposed to be complete but, in reality and in practice, it remains incomplete and inconsistent. Thus, it requires a reasoning and control strategy together with an inference mechanism. To the user or expert, the system interacts to collect more and more information at the time of execution. Interaction with the user requires some interfaces such as NLP, graphics or speech.

The entities such as knowledge base in the memory, the inference and the user's interface constitute the knowledge-base system.

AI IN THE FUTURE

In the special issue of JACM-January, 2003, three stalwarts (pioneers) in the area of artificial intelligence envisaged the direction and dimension of the movement of AI methodology for the next half a century. The areas of the development were: logical approach, non-monotonic reasoning, frame qualification, and interface design, semantic markers, and ultra intelligent computer. Following are the potential specific dots on the above topic of research.

- The ramification process involves the implicit specification that does not change when an event occurs, and the qualification problem involves elaborating sufficient conditions for an event to have its normal effect (Shanahan, 1997).

The major work in logical AI on facts about action and change is for discrete logic, i.e. logical approach which is concerned with the incorporation of humanlike AI and the methods to extract the humanlike performance to the machine, i.e. personification of machines.

- Learning from books, i.e. reading a text, extracting relevant information, the syntactic structure or semantics of the content, the content of discourse. Giving a programming languages primitive for the abstract syntax of the language itself.
- Continuous change and concurrent events, involving actions of several agents that require considerable efforts for future work.
- Other directions are 3D approximate knowledge for precise modeling, 3D robot's surroundings involving rectangular parallelepiped approximate real objects.

Interface Design of CS

The human–computer interface design involves computer graphics, sound synthesis, speech synthesis, speech recognition and haptics. These aspects of interface have already been developed and designed with the established methodology. But the mode of integration of chess functional entitles for human perception, cognition and factors that transform these bodily affairs to a minimum. It involves the physics and physiology of perception rather than the computational model and engineering design. The interface should have the following characteristics:

- It should be intuitive for the novice, efficient perception and notion of the expert and robust under misuse.
- It should facilitate in recovery from cognitive and manipulative mistakes.
- It should be helpful in diagnosis of errors and suggestive for corrective actions.
- In the early 1960s, the person who tilted the philosophical vision of artificial intelligence from theory to practice, was Edward Feizenbaum and also combined mathematics and medicine while researching at Stanford University. He pulled logic of mathematics for application to medicines-like domains where logic dominates computational mathematics. The reasoning process of experts in a particular domain was formalized and implemented in program, and there, in due course of time, knowledge-based system emerged as a framework that encompassed knowledge and reasoning.
- Feigenbaum proposed the following concepts: the ultra intelligent computer to solve the problems that were not reached by human yet were plausible.
- Machine learning, i.e. building a large knowledge base by reading texts.
- Development of resource description framework.
- Distilling from the www, a huge knowledge base, reducing the knowledge effort by the order of magnitude.
- Implementing ground vision, knowledge engineering must build a system of semantic scrapers that will access knowledge base, and set up material for scrutiny of the editorial process.

- Codify the knowledge by developing the volunteer built semantic net known as word net. Lenat (1983) in CLC to codify usable body of common knowledge.
- The development of complex computational engines such as human mind and brain.

APPLICATIONS OF AI

Following are the areas of AI applications.

In engineering, there are various disciplines such as civil, computer, chemical, communication, electrical, electronics, industrial, mechanical, metallurgy, mining and production engineering in which AI finds its wide and well accepted applications. We describe some of the areas or domains in each discipline where AI finds its scope in application for its algorithm, methods, tools and techniques.

Civil Engineering

There are some books and references which deal with expert systems in design, and structural design or architectural design that need the contribution of expert(s) in construction to commission of any building, highway or bridge. Computer-aided design (CAD) reflects primarily the design data in a generic manner where the data dominates the design knowledge. But in the knowledge-based systems or expert systems, programs consist of the views and solution strategy of the experts in soil mechanics, structure, architect, engineering and contractors. It is difficult to find all the experts at different times for their opinion at users who are engaged in the construction of an entity of social/public/individual use. The expert systems bring together, for any time to use, expertise in different subdisciplines of civil engineering.

Water resources management is another subarea of civil engineering that focuses on the use of water through proper management and control policy in colony, municipality, and industry. Above all, the resource is through river, canal and tanks, and reservoir. The utilization, storage (reservoir) and distribution of water needs a knowledge-based expert system to meet the requirement of different kinds of uses from rural (for irrigation) to urban (for drinking) having a large and complex network. Even the environment science to control pollution of different kinds needs a comprehensive approach by experts of different fields. Thus, it provokes the development of knowledge-based system.

Computer Engineering

Computer science and engineering is concerned with the design and development of computing systems, capable of performing the tasks for which human efforts are fruitless in specific time and unacceptable with accuracy. Broadly, AI is integrated with the design of hardware-based system, their fault tolerant increased capacity, scheduling and load balancing of parallel computer multiprocessors. One of the emerging and broad areas of AI in computing is its integration into software engineering.

The knowledge-base systems are being widely used in reusability, module verification, requirement analysis, functional design and program validation. The expet systems have been developed for determining the computer configuration according to the user's requirement

(XCON), scheduling and person loading of software development projects (COCOMO1), guiding managers in shaping the performance of subroutines. Other expert systems are MASK: assists help-line personal diagnose user problems for complex software program. Component-based design and design patterns are recent trends for AI to SE;PRESS debugs operating system software; IPT: Intelligent Peripheral Troubleshooter.

There are numerous expert systems developed software, which assist in most of the discipline and domain of computing systems. A brief review is presented in Chapter 12.

Industrial Automation and Manufacturing

Industrial Automation (IA) is concerned with introduction, incorporation and intervention of automation, starting from raw material handling, planning and production of items and their assembly leading to higher capable equipment and their inventory. Automation means less human interference, more computing with knowledge enriched environment. Flexible manufacturing system (FMS) has emerged to tie up with AI for industrial automation in most of the leading and pioneering industries in the world. Some of the expert system-based flexible manufacturing systems have been mentioned and elaborated in Chapter 19.

Important expert systems for the purpose are: EXPERT PROB allows factory workers to perform quality control tasks; DISPATCHER Selects transports and delivers parts for assembly while maintaining inventory records; FAISI: Helps to create optimum schedules; WELD: SCHEDULER/SELECTOR Helps in the selection of proper welding procedures and welding electrodes.

Equipment Maintenance

Detection and diagnosis of faults, repair and proper maintenance schedule, replacement policy are the basic tasks in the maintenance of equipment. This requires knowledge gathering from different resources as well as proper methodology for fault computing in any equipment. Many expert systems have been developed for troubleshooting different kinds of equipment. In the mechanical equipment, the expert systems are: PUMPRO diagnoses problems in centrifugal pumps; TUROMAT diagnoses vibration problem in large turbomachinery. RED (Rotating equipment diagnostic) diagnoses unusual vibration in rotating equipment; MENTOR (preventive maintenance) needs a large central air conditioner. In the telephone industry, the diagnostic expert systems are: ACE (automated cable expertise) analyzes phone company repair data and identifies area for preventive maintenance and further repair; COMPASS assists in telephone system maintenance by analyzing operating data and recommending maintenance actions; IDEA assists telephone technicians in diagnosing problems with a LAN; BDS troubleshoots a large signal switching network, base band distribution system; GAMSTTA diagnoses fault in telephone trunks.

For electrical system the expert system is TOGA. It analyzes insulation oil to diagnose faults in large utility transformers.

Management and Finance

Locating (dotting), generating and distributing resources; material, man, and money, is management. Of course, there are other issues, dimensions and disciplines of management

which are not of prime importance for this book. Various expert systems have been developed to address many facts and interdisciplinary issues of management. Chapter 21 deals with the interaction of expert systems or knowledge-based system to management and finance in detail. Here we mention some of expert systems for the purpose such as the MANAGER ADVISOR: It assists corporate managers with business planning; LENDING ADVISOR assists credit managers in analyzing commercial loan applications and structuring appropriate loan packages.

PLAN POWER helps the financial institutions in analyzing the personal needs of the clients to offer investment or other financial advice. It is implemented in LISP, basically an object-oriented model. CANAM TREATY advises on legal aspects of international trade transactions. EXPERTAX assists in corporate tax planning using common LISP. CFA: (Corporate Financial Advisor) aids in corporate financial planning using LISP. FEADV SYS: Foreign exchange advising system assists the foreign currency investors on trading decision using ART (an expert systems shell). INGOT Helps in financial forecasting using FORTRAN SALESTAX ADVISOR extends advice on the sales tax status of financial transactions for advertising agencies, commercial artists and designers using fix online class expert systems shell.

Office Automation

The basic aim of the office automation is not only to provide computer as an office aid but also a media for knowledge dissemination to lateral and lower level of colleagues and employees as well as a knowledge acquisition system from higher and lateral levels to the required level. For the very purpose that any particular level of hierarchy, transfer of knowledge makes the personnel at any node, in particular to take decision and inform his boss or subordinate at any node in the horizon or any level on the hierarchy. Nowadays networking in general and information technology in particular, www makes possible the knowledge transfer and interface in an office. However, data and knowledge trade-off requires a paradigm for their integration and conversion for the purpose of office automation. The development of expert systems in this direction is Letter of Credit Advisor: It is a small rule-based system developed by Helix (UK) to assist clerical personnel in preparing and paying letters of credit. CV FILTER helps the user to decide whether an applicant should be interviewed; developed on Expert Edge (Helix ES). Data classifier helps the user to classify the value of their data, developed on Ex sys (Ex System Corporation).

Robotics

Robotics is one of the prime areas of AI applications as shown in the 3D view of AI (Figure 1.1). AI methodology is applicable to robotics in two ways: One is design and control of robot, and the other is application of robots to various fields such as manufacturing, mining, medicine (surgery). The design of various types of robots or manipulator is primarily of mechanical engineering concern, but its control needs an interdisciplinary approach such as electrical engineering concepts for design of special purpose electrical machines: super motor or d.c. motor or micromotors, sensors for feedback; power electronics for implementing controllers. Other aspects such as robotics vision require electronics and communication engineering for building mobile robots with vision modems for remote control, high speed image processor and

adapter. Design and development of multi-robot working in cooperation and coordination with the perception and pervasiveness like human (humanoid robot) is a very challenging problem for AI or knowledge-based system approach. Humanoid robots imitating many activities of human motor actions, as well as emotions, pose many challenging facts to AI community.

Robot path planning and movement in collision free environment deploy some heuristic search technique such as A* and Iterative Deepening Algorithm (IDA*).

The application of robotic system, in places where human movement is restricted as in nuclear power plant or mines, works intelligently in such places and atmosphere where hazards prevail for human with high probability, robots need incorporation of intelligence for learning, self-organization and adaptation. Reasoning with uncertainties and mechanization (mechanical or automatic reasoning) enhances the intelligence in robot.

Nowadays, very interesting but complex phenomenon in robotics application is in game playing. For the last few years in Seoul Robotic Football (RoboCup) game has been played continuously which deploys a good quality and high calibre of knowledge-based system methodology.

Robotics in surgery is another surprising event in medical application that interacts with the expertise from medical and AI community, respectively.

Medical Computing and Informatics

The first and firm application of AI was the design and development of the expert system named MYCIN at Stanford University by Shortliffe and Buchernan in the mid-1970s. Many expert systems in different fields of medicine were developed. Among them are INTERIES: a consultant for general medicine, PIP: Present Illness Program, VM: The Ventilator Manager, PUFF: Pulmonary function, CASNET; Causal Associative Network to represent the pathogenesis of a disease, in terms of patient's findings. Many other expert systems have been developed for detection and interpretation of diseases depicted in the bioelectrical signals such as ECG, EMG, EEG with the advent of information technology, the telemedicine has marked a very important role in bringing IT, medicine and AI at one platform.

The impact of AI on hospital system (H/S) can be seen in developing object-oriented model of different aspects of H/S such as patient monitoring system, drug delivery system, hospital administration, ward management and clinical testing (pathological, microbiological, radiographic images (CT and ultra sound, MRI), and signals (EEG, ECG, and EMG). The most effective contribution of IT and AI together is the telemedicine and robotics in surgery.

E-services

E-services are concerned with the performing business, e-commerce, governess on www. E-commerce performs sales and purchase with customer relationship management (CRM).

Transportation

Transportation system is basically concerned with scheduling and planning. Scheduling departures and maintenance involve manipulating the relationships between certain objects, namely vehicles, destinations, and service facilities. Addition and subtraction of selected

change in routes are occasional but more attention is paid for maximizing the profitability of the current fleet serving current routes. The expert systems for the purpose are: SEATES assists analysts in adjusting the number of discount seats available on airline routes; AALPS configures air cargo shiploads; NAVEX monitors control on space shuttle flights.

Agriculture

Producers and traders of agro goods are main benefactors of expert systems applications in agriculture. An expert system assists farmers to produce crops, troubleshoot the crops once it is on ground and when to harvest it. Commodities loading is a complex business, which requires the expertise in understanding the marketing pattern, production of the grain, and distribution profile of the same. Some of the important expert systems in this area are WHEAT COUNSELOR. It is used for two purposes: buying guide for farmers shopping for agrochemical, and as sales aid for chemical manufacturing sales people; AQUAREF: Reference library for agriculture as a front end. PLANTING: It gives advice for various planting equipment. PGMA: It assists farmers to select the best way to market their grain.

Oil Exploration, Minerals and Metallurgy

The extraction of minerals and their metallurgical processing involves many phases and expertise ranging from geologists, material technologists, metallurgical engineers. Similarly, oil exploration requires expertise from geology, petroleum and chemical engineering groups. Following are the expert systems in practice.

DRILLING ADVISOR: A knowledge-based system developed by a French oil company that assists oil rig supervisor in resolving and, subsequently, avoiding problem situations. MUDMAN diagnoses problems with "mud" used in oil well drilling and recommends new compositions. WAVES aids in developing data processing control strategies for seismic survey data.

Electrical Engineering

The various areas of electrical engineering that require the knowledge-based system approach are:

1. **Power system:** Generation, distribution, load flow analysis, load management, power protection, power system control, high voltage transmission and distribution.
2. **Electrical machines:** Design of various motor alternators and generators needs a knowledge-based approach rather than computer-aided design (CAD), because consideration of different types of loads (primarily modeled as R, L, C and their combination) and drives (mechanical or electrical) pose uncertainties in the design parameter (change) and performance (degradation).
3. **Control:** Depending on the plant (object to be controlled) characteristics, the controllers are designed to have a desire performance of the plant. The plant may be a motor, robot, or a mechanical system. The basic controllers are Proportional (P), Derivative (D), Integral (I), PI and PID. Depending on the characteristics of the plant, the controllers may be linear/nonlinear, deterministic/stochastic, continuous/discrete,

lumped/distributed, time invariant/variant. Most of the controllers do not require AI/knowledge-based system methodologies for their design but complex mathematical models, instead.

4. **Power electronics:** It consists of design and development of converter (from a.c. to d.c.) and inverter (from d.c to a.c.) and control of power using various techniques such as chopper control cyclometer (frequency control), PWM (pulse width modulation), recently ANN and some heuristic search techniques have been deployed for the purpose.

5. **Drive control:** This field combines electrical machine, drives and control principles. Thus, different experts from the respective fields are required to show knowledge, control, and management for design, development and diagnosis of problems in railways, paper and pulp industries, textile industries, hoist and mills. It emerged from the knowledge of different disciplines for its effective use. Important components of KBS used in such problems are knowledge representation, inference and uncertainty measurement and users interface (graphics). Some of the expert systems are hoist diagnosis developed by INSIGHT INC. for diagnosis of faults on hoist equipments. DELTA (diesel electric locomotive trouble shooting aid) developed by General Electric company assists in assessing maintenance needs and prescribes appropriate action to assist diesel locomotive maintenance personnel.

ORGANIZATION OF THE BOOK

This book has been organized into three basic units: Logic, Search Knowledge-based System, AI and knowledge-based system application domain. In general, the core concepts of AI are logical computation and knowledge. Other areas are natural language processing, vision, speech recognition, robotics, and pattern recognition. These have still not been fused with the AI core concepts. These are entangled to AI but not in the AI kernel. The rapid growth of awareness and usability of the AI formalism in engineering and management where logic dominates numeric computation. AI acquires a respectable position with a warm welcome in these applications. Industries were reluctant even to notice its importance before the 1980s. But rapid and widespread application of concepts and methodologies of knowledge-based system, particularly the development of expert system (ES) application in industrial automation, flexible manufacturing and robotics, has drawn the attention of industries. The various concepts, models and methodologies in each of these units are described as follows.

Unit 1: Unit 1 contains five chapters. Chapter 1 covers the basic definitions and illustrations of intelligence, artificial intelligence, knowledge, knowledge system with the origin of AI and the chronological development in the concepts of core and peripheral components that constitute and contribute to AI formalism up-to-date. The correlation between AI and knowledge-based system, the outline of domain of AI and knowledge-based system in the industries, research organization, educational and institutional have also been described.

The heart of AI is logic. The computational model of logic is the preposition and predicate. The syntax and semantics of the predicate calculus, i.e. formal method of knowledge

representation, the resolution method in inference mechanism are described in Chapter 2. It can be termed computation logic or logic of computation. Model and temporal logic are also discussed in this chapter. Chapter 3 contains search methods and related problems. There are more than hundred formal and heuristic search methods, and to cover all simultaneously is a problem, keeping in view the scope of this book. Only some heuristic methods based on believed search, e.g. breadth first, depth first, best first and hill climbing are discussed with some of the problems of AI origin such as puzzle and travelling salesman are quoted and illustrated as and when required. The problem reduction methods and some game playing problems are described in Chapter 4. The minimax problem, alpha–beta cut, the iterative deepening A*, the state of the art of game playing and the solution strategy are illustrated with the enumeration of their computational complexities.

Some books deal separately with AI languages, or mixed with other AI concepts, the functional and logical programming with their semantics, structure and construction. In Chapter 5 an attempt has been made to deal with the concepts and programming techniques of LISP and PROLOG. Another aspect which does not belong purely to AI community but has come close to AI programming methodology in due course of time is the object-oriented model (OOM) and programming.

Unit 2: Most of the prevailing modelling and computational concept of knowledge such as representation, reasoning, acquisition, evaluation, management, discovery are described in detail in this unit.

Chapter 6 discusses informal method of knowledge representation such as rule-based system (RBS), semantics nets (SN), Frame, conceptual graph (CG), conceptual dependency (CD) and scripts. The correlation among these is also mentioned in this chapter.

Reasoning involves automatic and probabilistic methods. Chapter 7 covers monotonic and non-monotonic reasoning based on default logic, closed word assumption predicate computation circumscription. Model-based and case-based reasoning, truth maintenance system are also discussed with examples in this chapter.

The uncertainty measurement in the imprecise and incomplete knowledge is performed by Bayes' probabilistic method, certainty factor and theory of evidence given by Dempster and developed by Safer in due course of time. The aspects of uncertainty handling are described in Chapter 8.

One of the important aspects of knowledge engineering is knowledge acquisition. Extracting knowledge from the human expert and feeding it in machine after refinement is a tedious task. Program kits and tools are developed to perform knowledge engineering, knowledge acquisition and machine learning which are interleaved and intermixed entities rather than separate one.

The machine learning methodologies such as version, space, explanation-based learning, learning through examples, analogical and induction are described in Chapter 9. The induction methodology such as ID3 and some of the knowledge acquisition tools such as MOLE, SOAR and SEEK are also described briefly.

Distributed AI captures multi-agent paradigm. Different types of agents, their characteristics, the specification of knowledge and social agents, blackboard architecture, cooperation, conflict resolution and agent-based language have been covered in Chapter 10.

Chapter 11 deals with the different methods of users interface design. Single and multimode modalities have been described. Expert shells for the design of knowledge-based system have been developed from the mid-1960s to up-to-date in the various fields of engineering, and medicine, hardly any field of human endeavour is left untouched by the realm of knowledge engineering. It is not difficult but unwise and wasteful effort to describe the structure and specification of ES shell or expert system in every field with the same magnitude of excellence. To be wise, the salient features of small size, medium size and large size of some expert systems together with comparative view of some of the important expert systems in certain areas of relevance are described in tabular form to have a comparative view in Chapter 12.

Chapter 13 deals with the knowledge discovery and KDD data mining. These two concepts are inseparable, in true sense of principle and practice as well. Clustering, graph based and association rules and other heuristic methods of knowledge discovery and data mining have also been covered in this chapter. The application of these concepts and methods has been shown for some problems. Some issues of web mining are also dealt with in this chapter.

Knowledge management is forum for research and education with an application to information management, industries and organization of finance and business concern are described in Chapter 14.

One of the use of orbital processing elements for AI paradigm is the written language, i.e. text. Language translation, understanding, information retrieval, document summarization, discourse analysis are the vital aspects of natural language processing. Conventional methods, particularly, syntactic and symmetric analysis with their structure and grammar are described in Chapter 15. An attempt has been made to cover the emerging concepts and methods, up-to-date for discourse analysis, information extraction and summarization of the documents with knowledge-based system methodology. Statistical, rule-based and exemplar-based machine translation methods are also dealt with in this chapter.

The design steps of knowledge-based system and selection of a particular expert system shell are knowledge-based system components for a particular problem domain are described in Chapter 16, the various phases in the development of knowledge-based system together with their comparative view to software engineering has also been discussed. The development phases such as requirement analysis, selection of experts, and selection of knowledge-based system components for prototype design, implementation of the design, its formalization, performance evaluation and maintenance strategy are dealt in detail.

Unit 3: Various concepts, algorithms, methods, tools and techniques discussed in the chapters have certain potential to solve problem of different levels of abstractions in some areas of computer, electrical, communication, mechanical, production and management.

There is a common interaction between AI and software engineering methodology, so far as the design and performance evaluation strategies are concerned. The knowledge-based system concepts are used in different phases of software engineering development such as requirement or specification analysis design and reusability. These aspects are dealt with in Chapter 17. Some of the works in knowledge-based system approach to the problems of software engineering are described briefly in this chapter.

The earliest and profound impact of knowledge-based system was in the area of medicine. Medical professionals use vast and diversified knowledge for the diagnosis or decision in

detection and treatment of a particular disease. Chapter 18 covers the topics of bioelectrical signal ECG, EMG, EEG detection and interpretation, medical decision-making, cooperative view of some of the expert systems in medicine and hospital information design. Although artificial intelligence in medicine (AIM) is a wide area of education and research, therefore, it is important to make a comprehensive view of these topics.

Industrial automations is concerned with planning, scheduling and executing of various processes or jobs using different resources (machines) in certain quantity with specified quality and stipulated time span. In automatic machines, robots are used alone or in conjunction with machine for speeding up automation processes. In Chapter 19, flexible manufacturing systems covering planning and scheduling of processes and robot-based assembly, robotic controls are described with some specific issues of cooperation and collision-free path planning with knowledge-based system approach in multi-robot environment have been addressed.

Network management deals with the design of various network topology, selection of routing algorithm, capacity planning, detection of faults. The knowledge-based system approach finds relevant and encouraging scope to apply its tools and technique to the above-mentioned problems.

The application of knowledge-based system for some electrical and communication problem has also been discussed in Chapter 20.

Although a different non-engineering but fruitful and attracting course for engineering graduate is management, the management in industrial set-up, business organization and financial field covers a comprehensive level knowledge stream. Keeping these points in view, Chapter 21 introduces principles of management and practice in the fields mentioned earlier. The general architecture of knowledge-based system with their functional correlations to one another field of management science is described in this chapter.

In general, each chapter starts with introduction followed by content, summary, state-of-the-art (in some chapters) and exercises. The state-of-art is basically a research issue rather than educational content. Keeping in view the research thrust of graduates and research students, survey of the recent research directions are dotted in the state-of-art section in some of the chapters.

EDUCATION AND RESEARCH IN AI

Many centres, organizations and institutions are associated worldwide with the activities in research and education in AI. Although it is a difficult task to make a comprehensive and comparative view, an attempt has been made here to enumerate the activities of certain pioneering institutions in this area.

At MIT [CSAIL, 2002], USA, a group of persons is actively engaged in computer science and artificial intelligence laboratory (SAIL) earlier known as AI lab., in the areas such as machine learning from partially labelled data, on the Dirichlet prior and Bayesian regularization, reinforcement learning for dynamically stable legged motion, and neural voting machines.

Pattern Recognition and Vision

The working areas are feature selection for (nonlinear) regularized least-squares classification

and generalized robust conjoint estimation, learning object boundary detection from tracking data, video puppetry and automated feature induction for that classification.

Financial Applications

Multiple market-Markov behaviour in artificial financial markets and input–output hidden Markov models for modeling stock and/or flows are also the non-engineering fields.

University of Edinburgh [AQR: 2003]

It is one of the earliest prime centres of AI education and research in UK and Europe. The main research area is approximate and qualitative reasoning which is concerned with the development of techniques for representing common knowledge of the average person and the taut knowledge used by engineers and scientists. This is achieved by incorporating the common knowledge within formal and domain independent algorithm to perform reasoning about domain-specific problems.

Centres in India

In the 1980s five nodal centres were established to pursue different activities in the knowledge-based system. The DoE is the nodal centre of the KBCS (knowledge-base computer systems) projects whose objective is to develop AI applications in import policy, pension rules, etc. KBCS activities of C-DAC are concerned with the development of graphics and intelligence-based systems technology (GIST) for language processing, Indian language processor, particularly Sanskrit and Panini grammar auto-immunization, symbolic language programming environments in Devanagari. At IIT Madras, another centre aims at the following activities: Development of intelligent information systems for non-urban usage, expert system for engineering applications and development of expert system tools in R&D. The Indian Statistical Institute (ISI) Kolkata is also one of the centres, engaged in knowledge-based system approach, applied for image processing, pattern recognition, computer vision, approximate reasoning and machine learning. At National Centre for Software Technology (NCST), the specific activities are in the areas of intelligent information requirements, machine translation, intelligent testing and intelligence scheduling of ships and aircraft. VIDWAN, an expert system shell is its product. The centre at IISc Bangalore is concerned with the development of parallel architecture and supercomputer.

An article in CSI communications (Ananthakrishnan et al. 2006) presented an overview of the activities in AI in general and NLP, speech recognition and robotics in particular. Table 1.3 shows the various activities of the centre/institute and the responsible personnel.

Other Centres

At Stanford University and CMU, the AI active areas are also of prime importance. The centre at Stanford University is concerned with the development of mental model for multi-agent paradigm, robot path planning, geometric reasoning and vision. At the CMU, the centre is widely known for research group in the areas of robotics, autonomous vision system, mobile

Table 1.3 Institutes/personnel and their activities

Institutes/Personnel	*Activities*
IIT Guwahati Shivashankar B. Nair	• Speech to text: Interface cascaded with natural language processing for conversion of robot comprehensible command • Robin: Robots in the network to share intelligence over the network through intelligence access point
IIT Kanpur Amitabh Mukherjee Harish Karnik R.M.K. Sinha P.K. Kalra S.C. Srivastava	• Robotics: Qualitative models of reqrasping, Soccer playing robots vehicle motion planning • NLP, vision and speech recognition • Machine translation, Sanskrit parser, English to Hindi translation, the multilingual GIST technology, • Automated and common sense reasoning, KBS development • AI methodology for power system planning and control
IIT Hyderabad Rajeev Sangal Tata Consultancy Services limited Mumbai PVS Rao	• NLP: Language understanding Hindi, Telugu, English, machine translation speech processing their product SHAKTI translates English to various Indian and other languages • Bilingual, monolingual, lexical resources • Robotics: Research on multirobot system, planning, cooperation, navigation, multi-sensor surveillance • System for multi-target detection, for a feedback legged wheeled vehicles • Speech script and NLP robust speech recognition, language modelling, Knowledge-based NLP
IIT Madras Hema A. Murthy B. Ravindran Deepak Khemani Indian Statistical	• Automated speech recognition, Synthesis case-based reasoning • NLP and speech analysis on Indian languages
Institute Kolkata B.B. Choudhuri and others	

robots, machine learning and robotics and sensors (non-vision). The Robotics Institute (RI) at Carnegice Mellon University, founded in 1979, is the world's largest academic organization with the goal of making it the best place on the planet to do robotics research.

IBM

Knowledge discovery and data mining methodology deals with high performance scalable data mining technologies for large-scale databases and data repositories in its quest project. The exploratory vision group at IBM has performed studies how to use vision systems for applications to biometric, video browsing and automated object recognition supermarkets.

Deep Blue, one of the illustrious game play program which won over Garry Kasporov, the world champion, was developed by IBM, AI reason of group which relies on a combination of brute force and sophisticated knowledge of chess having capability of calculating 200 million

moves per second. Programs have been applied for computation biology, pattern discovery, data mining, optimization and bioinformatics, by integrating advances in algorithm, analytic methods and power of computing kernel. The IBM research group in AI is actively engaged in the area of knowledge discovery and data mining, human–computer integration, NLP, performance modelling and analysis and the AI-based application areas such as process scheduling and information technology.

SUMMARY

Intelligence, artificial intelligence, and knowledge are defined and the constituents and components of the knowledge-based systems are described.

- Intelligence is the ability to respond quickly and accurately.
- Artificial intelligence is the method of implementing the characteristics of human intelligence in machines.
- Knowledge is the mode of knowing an event at different levels of abstraction.
- The knowledge-based system is concerned with design, development and interaction of the systems, which exhibit the characteristics of human intelligence in solving problems of complex nature.
- The constituents and components of KBS are basically cognition and logic. The next level contributors are from the disciplines of psychology, reasoning, search, computation and information system. The three-dimension structure of AI consists of: logic-cognition and computation in x-direction, knowledge, reasoning, and interface in y-direction and application areas such as NLP, vision and robotics in z-direction.
- Various centres of AI research are IBM, MIT and, in India, five nodal centres of KBS.
- The future dimensions of progress in AI for the decades to come have been discussed in this chapter.

EXERCISES

1.1 Describe an application field for each of 3D model components in each of the dimension.

1.2 Describe the issues for using AI methodology in business intelligent and management application area.

1.3 Write about some of the centres in India and abroad where AI laboratory centres have been active for education and research in AI.

1.4 Write about some more areas or topics for research in AI in mechanical, civil, electrical, electronic, telecommunication, metallurgy, chemical and ceramics engineering.

1.5 Write about some more areas of research and education of AI in industrial automation.

Chapter 2



Logic and Computation

The development of logical theories and their contribution to modern computing science stems from Aristotle. There has been strides and renaissance from time to time. From the period of Aristotle-to-date, these have stirred the thoughts of logicians and turned the thinking pattern of educationists. In this chapter, we are going to deal with the development of logic theory to computation in chronological order, commencing from the Aristotelian period to the modern period, from Bertrand Russell to Godel. It is depicted chronologically containing the contributors, basic concepts and remarks.

The firm and ever pervading concept in AI development is logic.

CLASSICAL CONCEPTS

According to Bacon, logic is the science of correct use of intellect. It is a science but more abstract than any other sciences; its object being the instrument (origin) of other sciences and it investigates the conditions of all other sciences.

Rene Descartes (Renatus Cartesius, his Greek name, 1596–1650) divides philosophy into three parts: logic, physics, and mathematics (Principia Philosophy, 1644). Descartes developed the discourse method for conducting the reason with two principles and four rules. Two principles are as follows:

(i) **The methodical doubt:** From the methodical doubt a certitude emerges, which Descartes expressed in the famous enthymeme: "Dubito ergo cogito, cogito ergo sum" I doubte, ergo I think, I think, ergo I exist". Descartes drew conclusion from the discovery of this absolute truth that undoubtful truths are accessible to man.

(ii) The other is **clear and distinct ideas rising from this state of clarity and evidence of consciousness** or clear and distinct ideas to the rank of criticism of truth.

The four rules are:

- Never accept a thing as true, which has not been recognized obviously as true.
- Bring gradually complicated and obscure propositions to some simpler ones.
- Do gradual passage from simple to compound.
- Compute enumerations and general review to make sure that nothing was omitted.

Copyrighted material

The two main conclusions drawn from these rules are:

(a) Decompose things in their composing elements, the truth of which is given by intuition.
(b) Then from these composing elements, begin to recompose things through deduction, which goes from evidence to evidence and by means of induction in rising to more general truths.

Leibniz (1646-1716)

Logic has two assumptions according to Leibniz. These are:

1. Logic is a rational language for which Leibniz himself called the art of symbolic notations, which marvelously contract and summarize the mind's operations.
2. Logic is the encyclopedia of all knowledge conceived as an ordered systematization of all knowledge as a portable library which hence developed, in Leibniz conception, as a learning method.

This encyclopedia is magnum opus, which includes the following parts:

- Summary of all knowledge.
- All figures and tables, and schemes to illustrate encyclopedia.
- The collection of the literary thesaurus.
- The total amount of observations and experiences.
- The real method of inventing and reasoning which included the analytica and combinatorica.

About the truth, Leibniz stated the following principium:

(a) *Principium identitatis:* From which the analyticity of an idea, i.e. truth is established.
(b) *Principium contradictionis:* Through which it may be confirmed that any identical (analytical) proposition is true and its contradiction is false.
(c) *Principium reddendac rationis (The principle of sufficient reason):* Any true proposition, which is not known to itself acquire a proof *a priori* which means that it is possible to show reasons for any truth.

Immanuel Kant (1724–1804)

The great German philosopher developed a new theory of knowledge different from the prevailing one: empiricism knowledge senses (experience) and rationalism which states that knowledge is through reason.

In his fundamental work *The Critique of Pure Reason,* he stated that without sensibility we could be given no object and without intellect we would not think anything. Sensibility and intellect are two dissimilar faculties. The intellect can intuit nothing and senses can think nothing.

Kant defined logic as the science of intellect's rule in general. Kant developed a new kind of logic called *transcendental logic*, which appeared in *The Critique of Pure Reason* as

1. Transcendental analysis
   - Analytics of concepts
   - Analytics of principles
2. Transcendental dialectics
   - The paralogism of pure reason
   - Antinomies of pure reason
   - Ideal of pure reason

The two conditions are necessary to examine any knowledge. They are presupposed matter, and form. Therefore, he scrutinized, not the expression, where laws of matter are drawn from, but the cognizant spirit itself, to determine the form of knowledge. In order to demonstrate the possibility of existence of a prior knowledge, Kant started with the fact that all our knowledge is expressed by judgement of two kinds:

1. *Analytic judgement:* This analyzes a concept only in its composing elements without providing anything new.
2. *Synthetic judgements:* This provides an additional element of the concept, which is taken as the subject and from which, in no way, could we have drawn it.

The synthetic judgement is further divided into *judgement of expression* and *mathematical judgement.*

Kant showed a table of all kinds of judgements possibly, i.e. of all logical functions of the intellect within the judgement. The four kinds are possible:

1. By quantity	Judgement General Particular Singular	Category Unity Plurality Totality
2. By quality	Affirmative Negative Non-definite	Reality Negation Limitation
3. By relationship	Categorical Hypothetic Disjunctive	Inference and subsistence Causality and dependence Solidarity (reprocity) between active and passive
4. By modality	Problematic Assertional Apoidictic	Possibility–Impossibility Existence, Non-existence Necessity Contingency.

The first term of each group within the four groups of categories is a **condition**; the second term represents **conditioned**; the third, the **concept** resulting from the fusion of the

condition and conditioned. A close relationship exists between column judgement and column of categories; for instance, the cause category corresponds to the hypothetical judgement. Similarly, other correspondence can be derived.

The second part of transcendental logic, in the analytics of principles, Kant established the system of principles of pure intellect. The correspondence between the categories of each of the four groups and the principles of pure intellect is represented as follows:

Categories	**Principles of pure intellect**
Quantity	Axioms of citation
Quality	Anticipation of perception
Relation	Analogy of experience
Modality	Postulates of empirical thoughts in general.

The first two principles are called mathematical and the next two dynamic, because they are based on the existence of appearance.

HISTORICAL DEVELOPMENT OF LOGIC AND COMPUTATION

Table 2.1 shows the chronological development in classical logic and modern computation logic.

Tabel 2.1 Development in computation logic

Contributors (Logicians)	*Basic Concepts*	*Remarks*
Aristotle (Greek philosopher)	Syllogism: Concepts of propositional logic and FOL theory of implications, truth table.	Unable to reason out complex reasoning.
School of Megarian and Stoic logic. Mates (1953)	Five basic inference rules are valid without proof Modes Pones.	
Decrates (1641)	Discourse method for reasoning.	Distinguished logic from philosophy.
Wilhelm (1646–1716), (German philosopher)	Reducing logical interference to purely mechanical process without their realization.	The ideas were applied to formal languages.
George (1847)	The model of ordinary algebra of real numbers.	
Schroeder (1877)	Conjunctive normal form.	
Horn (1954)	Horn clause.	Used in logic programming prolog.
Third Earl of Stanphhope (1753–1816)	Theory of probability by compounding syllogism and certain inferences.	

(*contd.*)

Tabel 2.1 Development in computation logic (*contd.*)

Contributors (Logicians)	*Basic Concepts*	*Remarks*
Newell and Simon (1957)	Computer program for logical inference.	First group to present it in their work logic theories.
Wittgenstein (1953)	Method for testing validity of sentences in the language of propositional logic through Truth-tables.	
Whitehead and Russell (1934)	Rules of passage are used to move quantifiers to the front of formulae. The axiomatic style of exposition using some logically valid schema.	These appeared in their famous Principia Mathematic.
Skolem (1920)	Skolem constants and Skolem functions and general procedure for Skolemization.	Used in unification and resolution.
Turing (1936)	The validity of first-order logic was not decidable.	
Robinson (1965)	Use of propositionalization and Herbrand theorem.	
Gilmore (1960)	First program based on FOL.	

COMPUTATIONAL LOGIC

We are going to study Propositional Logic, First-Order Logic (FOL) Predicate Calculus, Symbol Tableau and Resolution Principle. In this context, certain terms are defined here.

Proposition: Proposition is a statement which may be true or false, but not both at the same time. When the language of proposition is associated with a deductive system of inference, it is called propositional logic.

Propositional calculus: This is syntactic part of logic which deals with the formation of complex propositions using a set of rules and connectives.

Propositional logic: Propositional logic deals with the validity, satisfiability, and unsatisfiability using equivalence laws. Validity, consistency and inconsistency of propositional logic, using truth table are the oldest approaches in propositional logic. The evaluation of the validity, consistency and inconsistency is an easy task. But the method is computationally impractical when the size of table grows exponentially due to increase in the number of logical symbols in the formula. The following methods are used for the purpose:

- Propositional logic
- Natural deduction and the axiomatic logic theorem
- Proof development using semantic tableau and resolution refutation
- Predicate logic



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

Applying term/variable pairs not used to modify s_1 may amalgamate two substitutions, s_1 and s_2.

For example:

$(F(x, y/z)\ (A/x, B/y, D/Z)$

becomes

$(g(A, B/D), (A/x, B/y, D/Z)$

The associative law is applied to substitution

$(B \cdot s_1) \cdot s_2 = B\ (s_1 s_2)$

and $(s_1 \cdot s_2) \cdot s_3 = s_1 \cdot (s_2 s_3)$

The commutative law is not permissible

i.e. $s_1 \cdot s_2 \neq s_2 \cdot s_1$

A unification pattern matcher will determine the consistent set of bindings for the variable in both patterns that results in identical patterns. The substitution that achieves a pattern match is called a unifier and the simplest substitution is called the most general unifier (mgu).

Unification: Theoretical Concepts

Suppose a set $\Sigma = Pf(x, z), P(y, a)$
A substitution Q is called a unifier for a set Σ if and only if ΣQ is a singleton set, a set having single element.

Consider $\Sigma = \{P(m, n, u), P(x, y, v)\}$

Consider the substitution $Q = (m/x, n/y, u/v)$

$\Sigma Q = \{P(x, n, v)\}$

Since Q is a singleton set, therefore, $Q = (m/x, n/y, u/v)$ is an unifier of Σ.

Two formulae α and β are variants of each other, if there exists substitution θ and σ such that

$\alpha = \beta\theta$ and $\beta = \alpha\sigma$

Consider $\Sigma = P(m, n, u), P(x, y, v)$ with two substitutions

$\theta = (m/x, n/y, u/v)$ and $\sigma = \{x/m, y/n, v/u\}$

$\Sigma = \alpha\theta = P(x, y, v) = \beta$

$\Sigma = \beta\alpha = P(m, n, u) = \alpha$

Most general unifier

Given two unifiable expressions α and β, there exists a unifier σ such that $\alpha\sigma = \beta$, then we say that σ is the most general unifier (mgu). If there exists any other unifier θ other than σ, then θ is an instance of σ. The most general unifier may not be unique. In other words, a unifier σ for a set Σ—$\{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n\}$ is mgu, if each unifier θ of Σ, there exists a substitution $\vee$ such that$\theta = \sigma o \cdot \vee$.

Disagree set

Two formulae α: $P(b)$ and β: $P(y)$ are not identical because of different variables, i.e. b and y, α and β, respectively. To unify them, the disagreement set is obtained and then eliminated. For the above formulae the disagreement set is (b, y). The disagreement set of Σ is obtained by comparing each symbol of all expressions in Σ from left to right and extracting from S the expressions whose first symbols disagree.

For $\Sigma = \{Pf(y), g(x),b\}, \{Pf(y), x, a\}$, the disagreement set $D(\Sigma) = \{g(y), b, a\}$

Unification algorithm: Following steps are required for the unification and to find out the mgu.

Step 1: Set $k = 0$ and $\sigma_k = \varepsilon$ (the empty set)

Step 2: If S_k is a singleton set
then σ_k is the mgu of S,
Unifiable = true; stop,
else
find the disagreement set D_k of $S\sigma_k$

Step 3: If there is a variable v and term t in D_k such that v does not occur in t,

Step 4: Put $\sigma_{k+1} = \sigma_k(t/v)$
and $k = k + 1$ and $S_k = S_{k-1} \cdot \sigma_k$,
repeat Step 2.

Step 5: Otherwise output S is not unifiable.

Example 2.6

$S = P\{f(y), g(x), b\}\ \{P(z, u, a)\}$

$\sigma_k = \varepsilon$, unifiable progress = true

$S_0 = \{Pf(y), g(x), b\}, \{P(z, u, a)\}$

$D_0 = \{f(y), z\}, \sigma_1 = z/f(y)$

$S_1 = S\sigma_1 = \{P(f(y), g(x), b)\}, \{P(f(y), u, a)\}$

$D_1 = \{U\}$

$\sigma_2 = \{\sigma_1 o(u/g(x))\} = \{z/f(y), u/g(x)\}$

$S_2 = S_1\sigma_2 = \{P(f(y), g(x), b)\}, \{P(f(y), g(x), a)\}$

$D_3 = \{b/a\}$

$\sigma_3 = \{\sigma_2 o(a/b)\} = \{z/f(y), u/g(x), b/a\}$

$S_3 = S_2\sigma_2 = \{P(f(y), g(x), b), P(f(y), g(x), b)\}$

S_3 is a singleton set, unifiable true.

PREDICATE CALCULUS IN PROBLEM-SOLVING

Example 2.7 The following problem illustrates the use of resolution principles:
If Tom eats whatever Jade eats, and Tom eats banana, then what does Jade eat?
The problem is split into facts at atomic level to make a set of WFF.

1. $(\forall x)$ {Eat (tom, x) $\rightarrow$ Eat (jade, x)}
2. Eat (tom, banana).

The answer to the question what does Jade eat, one has to prove that the WFF

$(\exists x)$ Eat (jade, x) follows from S.

As usual, to prove this we need to prove the negative of WFF, and then it is added to the negative set. All the members of this enlarged set are converted to clause form. The resolution principle is applied for drawing the conclusion.

The negation of $(\exists x)$ Eat (Jade, x) is

$\exists x$ [~Eat (jade, x)]

And the clause form is ~Eat (jade, x).
The refutation tree is shown in Figure 2.3.

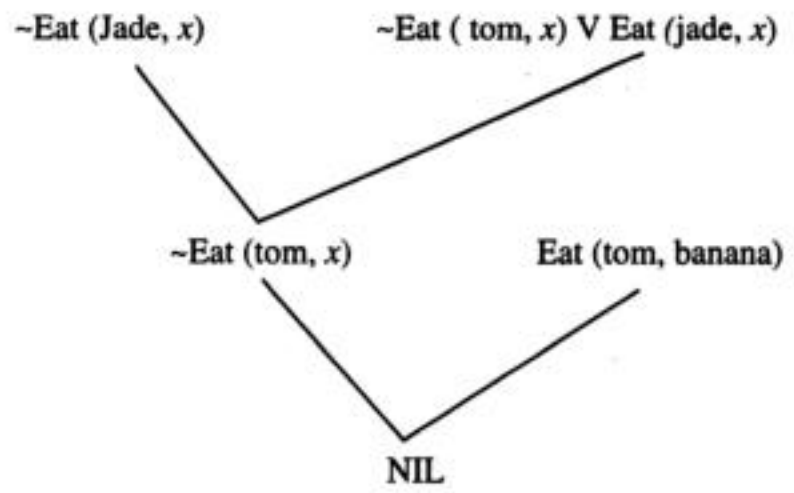


Figure 2.3 Refutation tree for Example 2.7.

The answer is obtained with transforming the clauses to its tautology as shown in Figure 2.4.

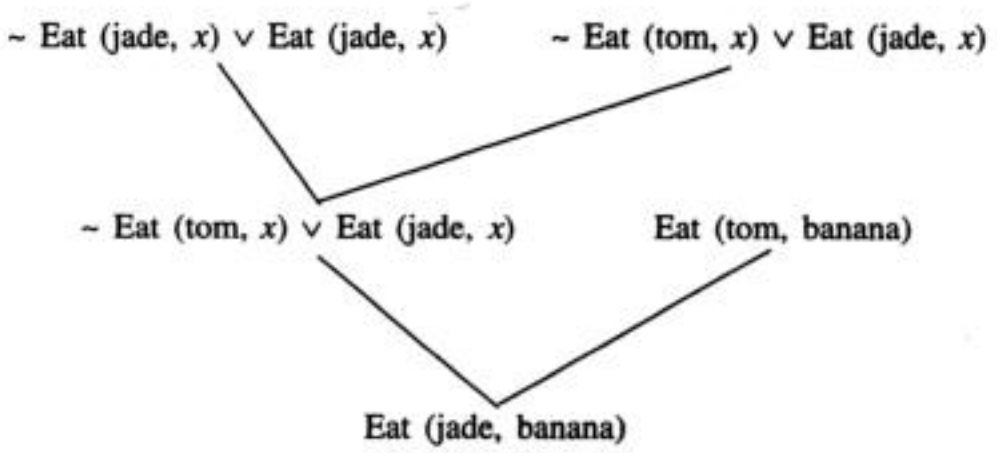


Figure 2.4 The modified tree for Example 2.7.

Example 2.8 All teachers are good. Anyone who is good and intelligent will deliver excellent lecture. Mohan is an intelligent teacher, shows that Mohan will deliver an excellent lecture.

The above text can be translated into symbol and predicate as follows:

$P(x)$ = x is a teacher.

$Q(x)$ = x is good.

$I(x)$ = x is intelligent.

$R(x)$ = x delivers an excellent lecture.

$U(x)$ = Mohan will deliver an excellent lecture

The corresponding resolution tree is generated as follows:

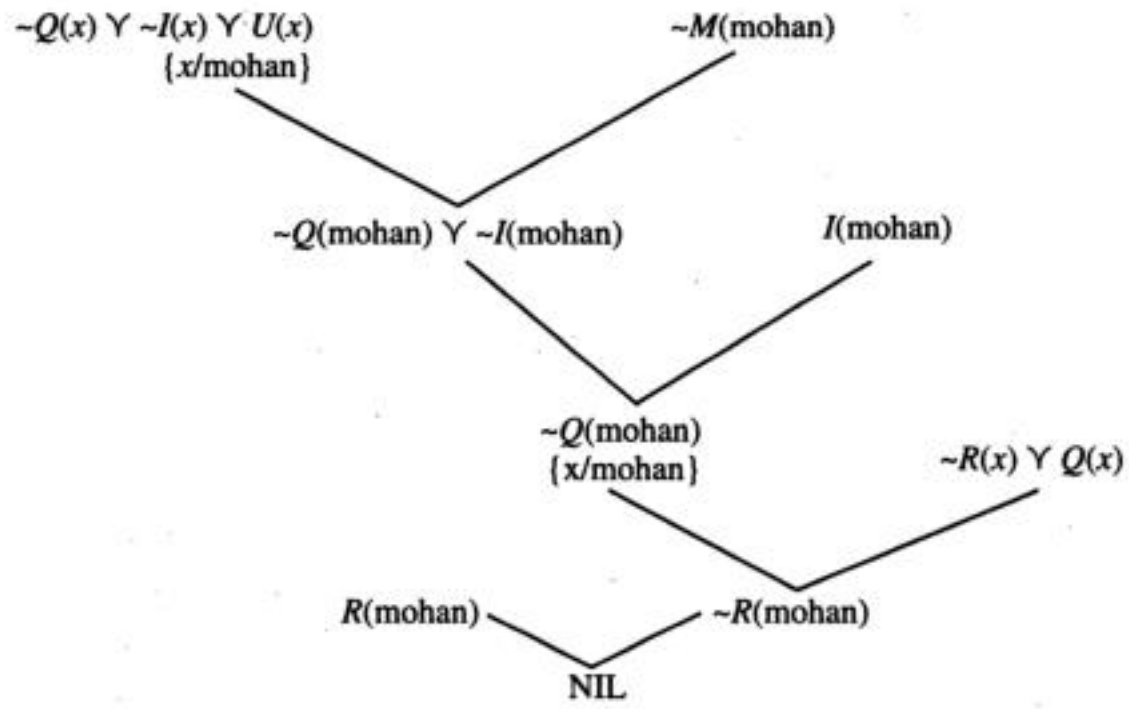


Figure 2.5 Refutation tree for Example 2.8.

We obtain nil with the coordination in contradiction to a set S U (D) which shows that U is a logical consequence of S.

Proofs of theorems

The theorems of a system are those wff which can be derived from their axioms by applying their transformation rules. To prove a theorem is, therefore, to derive it in this way. More precisely, a proof of a theorem α in a system S consists of a finite sequence of wff, each of which is either (i) an axiom of S or (ii) a wff derived from one or more wff occurring earlier in the sequence, by one of the transformation rules or by applying a definition, α itself being the last wff in the sequence. (Note that by this account of what constitutes a proof of a theorem, every wff in a proof is itself a theorem; and also that one reason why we count the axioms themselves as theorems is that any axiom can be thought of as a one-line proof of itself.)

Consistencey

We will say that an axiomatic system is consistent iff not every wff is a theorem of that system. In other words, a system is inconsistent iff every wff is a theorem.

MODEL LOGIC

The classical concept of interpretation of formulae is extended to the model logic such that it associates with itself, not only a single interpretation but also a set of interpretations called words. Similar to the interpretation, the formula is classical logic, a truth value is assigned to a formula in each word.

The model logic is defined by

$$V\colon F \times W \rightarrow (T, \perp)$$

where, F is the set of formulae of the model theory and V assigns a truth value to every formulae in F in every word W.

It is denoted by $< W, R, V >$ where W is the set of words; R is subset of $W \times W$ is the readability relationship between words, and V is the evaluation function for the formulae.

The relation R is characterized with relational properties such as reflexive, transitive and symmetric.

Reflexive: All the words are possible with respect to themselves, i.e. if $(W_j\, W_j) \in R$ for each $W_j \in W$ then R is reflexive.

Transitive: If when $(W_i, W_j) \in R$ and $(W_j, W_k) \in R$ then $(W_i, W_k) \in R$. Then R is transitive. If W_j is accessible from W_i and W_k is accessible from W_j, then W_k is accessible from W_i.

Symmetric: R is symmetric if when $(W_j, W_i) \in R$ then $(W_i, W_j) \in R$, i.e., when W_i is accessible from W_j then W_j is accessible from W_i.

When R is reflexive, transitive and symmetric it is called *equivalence*. Unlike classical logic or FOL, the model logics concerned with different 'models' on which a statement may be true. The different models of statement or assertion may be beliefs, desires, intentions which may be unjustified, irrational, or mutually contradictory. It provides a powerful tool for understanding the expressiveness of natural language, which often involves reference to other terms and circumstances.

Model logic introduces two operators necessary and possibly denoted by L and M respectively, apart from the operators and symbols of classical logic. The concepts of the necessity and possibility of formulae are expressed in the set word in which the main formula is evaluated.

Semantics of Logic and Model Axioms

The semantics of logic is given on the basis of evaluation function V which is inductively defined over the structure of the formula to evaluate.

V is defined over the model operators L and M as follows:

$$\neg V\,(Mf, w) = T \text{ iff } \exists V \in W \cdot w\, R\, v \rightarrow V(f, v)$$

It means that the formula Mf is true in word W if and only if there exists a word v which is reachable from W, where subformula is true.

$$\neg V\,(Lf, w) = T \text{ iff } \forall V \in W \cdot w\, R\, v \Longrightarrow V(f, v)$$

It reads the "formula Lf is true in W if and only if in all words reachable from W, sub formula f is true" M and L are existential and universal quantifiers, respectively.

The following relation holds between L and M:

$Lf = \neg M \neg f$

It states that possibly the formula f is equivalent to the statement that it is not necessary that there is no formula f.

The other model axioms are

$Lf \rightarrow f$ (if formula f is necessary, then formula is necessary)

$Lf \rightarrow LLf$ (if formula f is necessary, then formula is necessary)

Lewis axioms: Lewis formulated five systems of model logic of increasing strength $S1$ to $S5$. defining a connective $\Rightarrow$ for strict implications.

1. $(p \wedge q) \Rightarrow (q \wedge p)$
2. $(p \wedge q) \Rightarrow p$
3. $p \Rightarrow (p \wedge p)$
4. $((p \wedge q) \wedge r) \Rightarrow (p \wedge (q \wedge r))$
5. $p \Rightarrow r \vee p$
6. $((p \Rightarrow q) \wedge (q \Rightarrow r)) -3(p \Rightarrow r)$
7. $(p \wedge (p -3q)) -3q$,

where p, q, r are propositional variables, and the following rules of inference.

Rules of inference

- Uniform substitution of formulae for propositional variables.
- Substitution of strict equivalents: from $(\alpha = \beta)$ and γ infer any formula obtained from γ by substituting β for some occurrence(s) of α.
- Adjunction: from α and β infer $\alpha \wedge \beta$.
- Strict detachment: from α and $\alpha -3 \beta$ infer β.

System $S2$ is obtained by adding the axiom $\Diamond(p \wedge q) \Rightarrow \Diamond p$ or $(p \Rightarrow \Diamond p) \Rightarrow (p \Rightarrow q)$ to the axioms for $S1$. $S3$ is $S1$ plus the axiom $(p \Rightarrow q) \Rightarrow (\neg\Diamond q -3\neg\Diamond p)$. $S4$ is $S1$ plus $\Diamond\Diamond p \Rightarrow \Diamond p$, or equivalently $\Box p \Rightarrow \Box\Box p$. $S5$ is $S1$ plus $\Diamond p \Rightarrow \Box\Diamond p$. Here $\Diamond$ and $\Box$ are 'possible' and 'necessary' terms for model logic.

Kripke model

Kripke model is a tuple in a frame $<W, R, V>$ where W is a non-empty set of a possible word, R is the accessibility relation, and V a valuation function that assigns to each atomic sentence p a set of words. These models allow the use to define the model theoretic notions of truth, logical truth, and logical consequence.

A Kripke model is a tuple $<W, R, \|\!\!-\!\!>$, where $<W, R>$ is a Kripke frame, and is a relation between nodes of W and modal formulae, such that:

- $w \|\!\!- \neg A$ if and only if $w \not\|\!\!- A$,
- $w \|\!\!- A \rightarrow B$ if and only if $w \not\|\!\!- A$ or $w \|\!\!- B$,
- $w \|\!\!- \Box A$ if and only if $\forall u$ $(w R u \Rightarrow u \|\!\!- A)$.

the relation $\|\!\!-$ is called the *satisfaction relation*.

Illustration: For a problem in belief, information acquisition, and trust (BIT) problem, the Kripke semantics is utilized as follows:

$B_i\phi$ is the belief agent i has for the fact or formula ϕ.

$I_{ij}\phi$ is the information acquired by i from j for the formula ϕ.

$T_{ij}\phi$ is the trust for the justified information acquired by agent i from agent j for formula ϕ.

The formal semantics for B_i and I_{ij} is the Kripke semantics for normal model operators, whereas that for T_{ij} is the so-called minimal (or neighbourhood) semantics. Formally, a BIT model is a tuple $(W, \pi,(B_i)_{1\pi1\leq i\leq n}, (I_{ij})_{1\leq i\neq\leq n}(T_{ij})_{1\leq i\neq\leq n})$, where

- W is a set of possible words,
- $\pi : \Phi_0 \rightarrow 2^W$ is a truth assignment mapping each atomic proposition to the set of words in which it is true,
- $B_i \subseteq$ WXW is a serial, transitive and Euclidean binary relation on W,
- $I_{ij} \subseteq$ WXW is a serial relation on W,
- $T_{ij} \subseteq$ WX 2^W is a binary relation between W and the power set of W.

Godel on Provability as a Modality

Godel (1931) reviewed Becker's 1930 article. In reference to Becker's discussion of connections between modal logic and intuitionistic logic he wrote:

It seems doubtful, however, that the steps taken here to deal with this problem on a formal plane will lead to success.

Godel proposed the theory of completeness and also formalized assertions of provability by a propositional connective B (from "beweisbar"), reading $B\alpha$ as "α is provable". He defined a system which has, in addition to the axioms and rules of ordinary propositional calculus, the axioms

$Bp \rightarrow p$,
$Bp \rightarrow (B(p \rightarrow q) \rightarrow Bq)$,
$Bp \rightarrow BBp$,

and the inference rule: from α infer $B\alpha$. He stated that this system is equivalent to Lewis' $S4$ when $B\alpha$ is translated as α.

TEMPORAL LOGIC

Temporal logic is the extension of classical logic built by a set of new operators that hide quantification over the temporal domain.

The evaluation instant of a formula is used in this logic.

G: always true;

F: eventually in the future;

H: always in the past;

P: eventually in the past;

These are formally defined as:

$$V\,(Gf,\ t) = T\ ,\ \text{iff}\ \forall\ s \in T,\ t < s \Rightarrow V(f,\ s);$$
$$V(Hf,\ t) = T\ ,\ \text{iff}\ \forall\ s \in T,\ s < t \Rightarrow V(f,\ s);$$
$$Ff = \neg G\ \neg f;$$
$$Pf = \neg H\ \neg f;$$

The concepts of necessity and possibility is expressed by (G, H) in the future and (F, P) in the past by these operators. These operators are often represented by other symbols □ (always) denotes G and ◊ (Eventually) denotes F, H and P for past operators.

Two other binary operators are introduced if relation < is transitive and non-reflexive.

Until represented by μ: With $\phi 1$ until $\phi 2$ that is true if $\phi 2$ will be true in the future and until that instant $\phi 1$ has been true.

Since represented by $\mathfrak{S}$: With $\phi 1$ since $\phi 2$ that is true if $\phi 2$was true in the past and since that instant $\phi 1$ has been true.

The semantics of these operators can be formally defined as follows:

$$V\,(f1\ \text{until}\ f2,\ t) = T\ \text{iff}\ \exists s \in T \cdot t < s \wedge v\,(f2,\ s) \wedge \forall u \in T \cdot t < u < s \Rightarrow V\,(f,\ u)$$
$$V\,(f1\ \text{since}\ f2,\ t) = T\ \text{iff}\ \exists\ s \in T \cdot t < s \wedge v\,(f2,\ s) \wedge \forall u \in T \cdot s < u < t \Rightarrow V\,(f,\ u)$$

These two operators express the concept which is not possible by G, H, F, and P.
These two operators are correlated with future and past operators as follows:

$F\phi = T$ until ϕ; $P\phi = T$ since ϕ;
And $G\phi = \neg\ F\ \neg\ \phi$; $-\ H\ \phi = \neg\ P\ \neg\ \phi$;

The operator 'until' is enough to compute the logic expressiveness when temporal domain is bounded in the past. Since is not necessary when the past is limited.

There are two other unary operators next and previous represented by O and defined in terms of until and since as:

$$O\ \phi = \perp\ \text{until}\ \phi;\ -\ O = \perp\ \text{since}\ \phi$$

Properties on Temporal Domain Structure

The relation R is called a precedence relation and is denoted by <. The main properties of temporal logic are related to the properties of relation R. Following are the properties of temporal domain structure.

Transitivity	$\forall\ x\ y\ z \cdot x < y \wedge y < z \Rightarrow x < z$
Non-reflexivity	$\forall\ x\ ,\ -\ x < x$
Linearity	$\forall\ x\ y \cdot x < y\ v\ x = y\ v\ y < x$
Left linearity	$\forall\ x\ y\ z \cdot y < x \wedge z < x \Rightarrow y < z\ v\ y$ $= z\ v\ z < y$

Right linearity	$\forall x\ y\ z \cdot x < y \wedge x < z \Rightarrow y < z < y$
	$= z \vee z < y$
Begin	$\exists\ x \cdot \neg\ \exists y \cdot y < x$
End	$\exists\ x \cdot \neg\ \exists y \cdot x < y$
Predecessor	$\forall x \cdot \exists\ y \cdot y < x$
Successor	$\forall x\ y \cdot x < y$
Density	$\forall x\ y : x < y \Rightarrow \exists z \cdot x < z < y$

A linear temporal structure that corresponds to the intuitive notation of time exists when the precedence relation is linear.

When the precedence relation < is only linear on the left, then it is called left linear. In this situation, branches can exist in the future but only in part. There is possibility of existing temporal structures with branches in the past known as right linearity. When the temporal states are bounded in the past and the future, these are denoted by 'begin' and 'end', respectively. Predecessor and successor stand for unlimited temporal domain in the past and the future respectively. If there always exists an instant between two instants, then a temporal domain is dense with respect to relation <.

Types of Temporal Logic

There are mainly two categories of temporal logic:

1. Which shows quantification on the variable time or without a metric for time.
2. With a metric for time.

In each category, there are different types of temporal logic: a brief description is given below. A detailed discussion is beyond the scope of this book. Some temporal logics are as follows.

Propositional Temporal Logic (PTL)

Harel (1977) introduced this logic by introducing the four operators □, ◊, 0, and μ. The temporal structure of PTL is linear, bounded in the past, unbounded in the future, and discrete. The temporal operators □, ◊, and μ correspond to G, F and until. The formula $O\ \phi$ is a valid formula if the formula ϕ is true in the next state. Time consists of a sequence of instants corresponding with the sequence of states of the system. The fundamental entity of the logic is the instant in which the state of the system changes.

Branching Time Temporal Logic (BTTL)

It was introduced by Ben-Ari et al. (1983) is an extension of PTL having branches in future in its structure temporal. It is equivalent to PTL both on different evolution traces and states, the four operators have been defined.

1. $V\Box$, for all traces π and for all states $S \in \pi$;
2. $\exists\Box$, for all traces π and at least one states $S \in \pi$;
3. $\forall\Diamond$, for all traces π and at least one states $S \in \pi$;
4. $\exists\Diamond$, for at least one track π and for at least one state $S \in \pi$.

Interval Temporal Logic (ITL)

It has been introduced by Halpern (1983) and further extended by Helpern et al. (1983). It is a proportional logic with a structure temporal that is bounded in the past, unbounded in the future, discrete and linear. It is event-based logic and the fundamental entity of ITL is the interval mode of a sequence of states. The length of an interval is defined as the number of states in the sequence. Apart from the previously defined operators O, $\Box$, $\Diamond$, there are other two operators defined as Chop and Len (n). The 'Chop' permits one to concatenate state sequences and Len (n) is used to count the number of states in a sequence which specify the exact duration in terms of the number of transitions among events. It is used in modiling of digital systems.

Interval Logic (IL)

It is based on time interval and propositional logic, this was introduced by Schwaitz (1983), having linear, bounded past and unbounded in future. The 'IL' formula has the following form $[\phi]\ ^*\alpha$, where α is formula and ϕ is the interval, that is, the context in which the formula α has to be verified. The bounded version of operators $\Diamond$ and $\Box$ is presented by Ⓐ. The operators at, in and the bound is defined by means of the interval: $[\phi]\ \Diamond\ \alpha$ means that α can be true in ϕ.

SUMMARY

The logic and computation go together in AI methodology of problem solving.

- Logic dwells on the principle of tautology, modes pones, proposition, predicates and reasoning.
- The first order predicate logic (FOPL) with it's syntax, semantics, clause form conversion as formal method of knowledge representation has been described.
- Resolution is one of the methods of inference in the FOL. The unification principle has been illustrated for variable substitutions in the inference mechanism.
- Modal logic for quantification of all and some predicates has been described in the FOL format.
- Temporal logic which takes into account the quantification of time is a prime concern in the real time system.

EXERCISES

2.1 Express the following statements in modus ponens:

(a) If x is true, x being true implies that y is true, then y is true.

(b) Every student reads the topic or ignores it.

(c) She is beautiful and intelligent but not sincere.

2.2 Convert the following sentences into conjunctive normal form:

(a) $(x \,\&\, \sim y) \vee (x \,\&\, \sim y) \,\&\, z$

(b) $z \vee (x \,\&\, y \,\&\, \sim z)$

2.3 Express the following sentences into predicate logic:

(a) Tom is a student of computer science but not a cricket or a baseball player.

(b) If you do not run, then you will be lazy.

(c) For all students, there is some time for play and watch a movie.

2.4 Formulate the following expressions to predicate logic:

(a) Mohan would be happy to donate a sum of $11 or $21.

(b) All persons either like it or dislike it.

(c) The two resistances can be connected either in series or parallel.

(d) Ammeter is connected to series and voltmeter in parallel.

(e) The combination business is to business and business to consumer and consumer to consumer is true.

2.5 Using the equivalence laws, change the statements whichever is possible in FOL in Exercises 2.4 into commutative, distributive and associative form.

2.6 Draw a semantic tree for the set of clauses:

(a) $P(x) \vee Q(y)$, $\sim P(a)$, $\sim Q(b)$

(b) $\{\sim P(x) \vee Q(y)\}$, $P(f(y))$, $\sim Q(f(y))$

2.7 Consider the following set of clauses and show the use of resolution principle on them using unification

$\{Li\} = P(x, f(a)) \vee P(x, f(y)) \vee Q(y)$

$\{Mi\} = \sim P(z, f(a)) \vee \sim Q(y)$

2.8 Discuss the unification problem and explain with example the term 'mgu' in this context.

2.9 (a) Find the refutation graph for the given sets of clauses.

$\sim P(x) \vee Q(x) \vee R(x)$

$\sim Q(x) \vee B(x)$

$\sim Q(x) \vee C(x)$

$\sim R(x) \vee B(x)$

$\sim R(x) \vee D(x)$

$P(f(x)) \vee Q(g(x))$

(b) It is required to prove from the above axioms the conjecture

$\exists\ x\ \exists\ y\ \{\{B(x) \wedge C(x)\} \vee \{D(y) \wedge B(y)\}\}$

2.10 What are two operators introduced in the FOL to interpret the model logic? Discuss their properties and semantics.

2.11 What are the operators used in temporal logic? How the concepts of necessity and possibility are expressed in it? Show some of its properties and enumerate its application in the real time system.

Chapter 3



Heuristic Search

Search is the process of finding a goal, i.e. desired solution from a given initial state of the problem. This process requires defining the state, operators and state space in the description of the problem. Basically, there are various heuristic search methods and different types of problems on which these methods are applied to get the solution. The problem state is the particular configuration or situation of the problem where an operator transforms one state into another. The state space is all possible configuration or situations of the states reachable from the initial state to the goal state. For example, in an 8-puzzle problem as given below, the state is the configuration obtained by different positions of the tiles. It may be one of the states and movement of the empty space blank, which can be performed with four operators left, right, up and down. The initial state and the goal state is shown in Figure 3.1, and all other states between the initial state and the goal state can be obtained by the simple movement of the empty space left, right, up and down. This is illustrated in breadth first method.

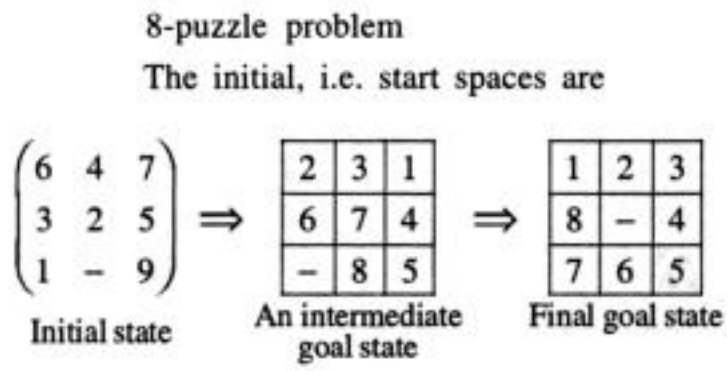


Figure 3.1 Initial state and goal state in a puzzle problem.

Various methods of search are broadly classified into two categories:

1. **Blind search:** Breadth first, depth first, and bidirectional search.
2. **Informed search:** Hill climbing, best first, branch and bound, and an optimal search.

The various problems tackled by AI methodology using these search methods are as follows.

SEARCH-BASED PROBLEMS

There are many problems, which require heuristic search methods for solution. These problems are 8, 15 and 24 puzzles, water-jug, tower of Hanoi, monkey banana and travelling salesman, and so on. A brief description of these problems with their state description is being given in this section and details of the solution strategy for some problems are given in the sections related to search methods.

Slide Bar Puzzle Problem (8, 15 and 24)

The 8, 15 and 24 are the puzzle problems of numbers in the cells to be arranged in a required format, from any given arbitrary format. The format arranges the numbers in a particular manner. In a search strategy, format is termed *state*. Some initial state and corresponding required state for 8, 15 and 24 slide bar puzzle problems are given below in their respective state space representation with cells filled with decimal numbers and a slide bar (–). The change of state can be performed by the movement of bar to its left, right, up and down positions depending on its particular position (Figure 3.2).

8 puzzle:

2	3	1	⇒	1	2	3
7	8	4		6	5	4
4	–	5		7	8	–

15 puzzle:

4	5	1	2	⇒	1	2	3	4
11	13	14	6		12	13	14	5
12	–	15	7		11	–	15	6
10	3	8	9		10	9	8	7

24 puzzle:

5	18	20	21	24	⇒	24	23	21	22	20
4	19	–	22	23		15	16	17	18	19
3	17	16	15	14		14	13	12	11	10
2	10	11	12	13		5	6	7	8	9
1	6	7	8	9		4	3	2	1	–

Initial states Final states

Figure 3.2 State space representation of the 8, 15 and 24 puzzle problems.

Tower of Hanoi

In the city of Hanoi, there were three pegs in the courtyard of a ministry. One of them contained 64 disks, placed in the order of lowering diameter from bottom, through the holes in the disk. The task of the monks was to transfer all the disks from one peg to another to be placed in the same order using the third peg as an intermediate temporary placing stand. The following conditions are imposed:

1. Only one disk can be moved at a time.
2. No disk should have above it, another disk with larger diameter.

This problem can be illustrated with taking three disks in consideration as shown in Figure 3.2.

The state representations from the initial state to goal state through various intermediate states are shown in Figures 3.4.

The states are described by three variables *A*, *B* and *C*; i.e. (*a b c*). (*a a a*) represents that all the disks, i.e. (1 2 3) are on peg *A* and (*c c c*) represents all the disks on peg *C* as shown in Figure 3.3.

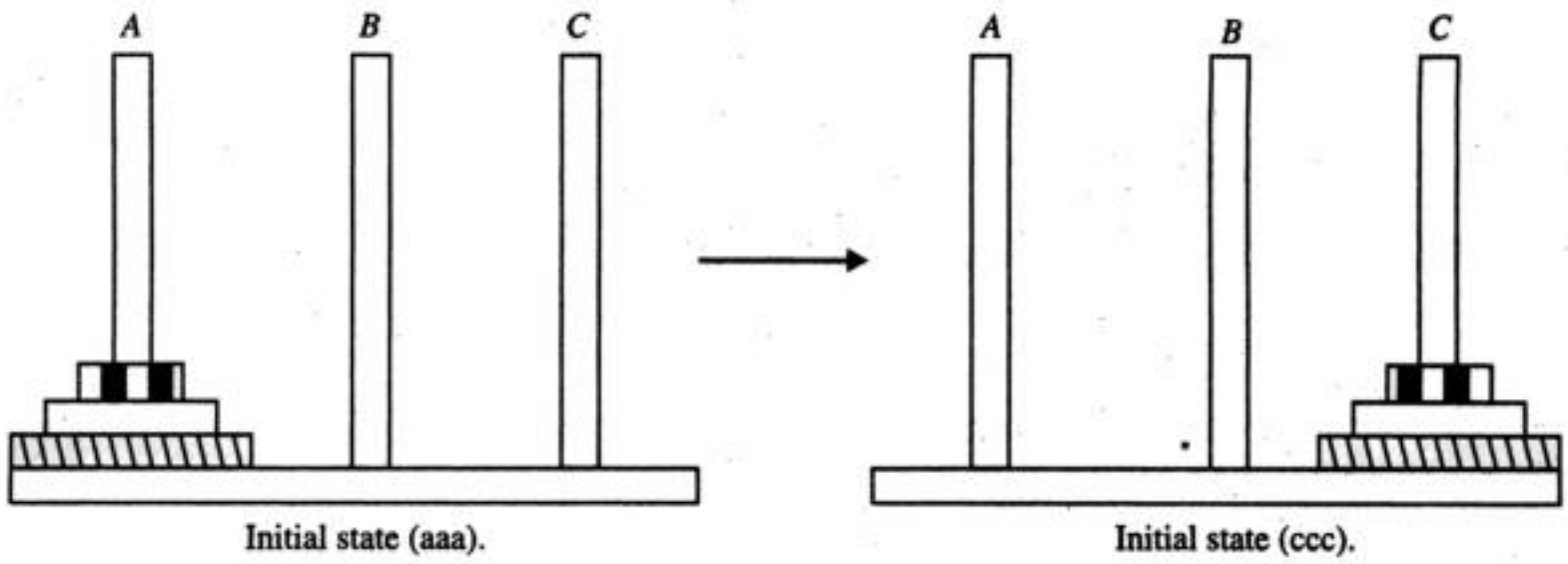


Figure 3.3 Initial and final state.

Figures 3.4(a) to (f) show respectively various states of the movement of disks as (*a a a*) ==> (*a b c*) ==> (*b b c*) ==> (*c b a*) ==> (*c c a*) ==> (*c c c*) respectively from the initial state (*a a a*) to goal state (*c c c*).

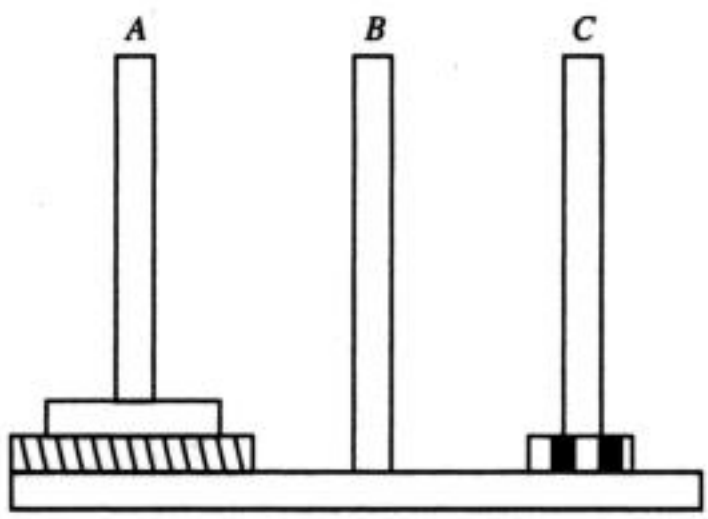


Figure 3.4(a) State (*a a c*).

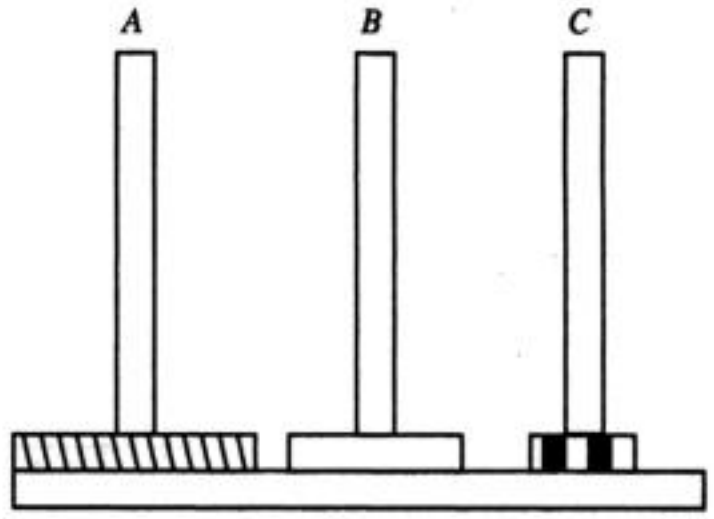


Figure 3.4(b) State (*a b c*).

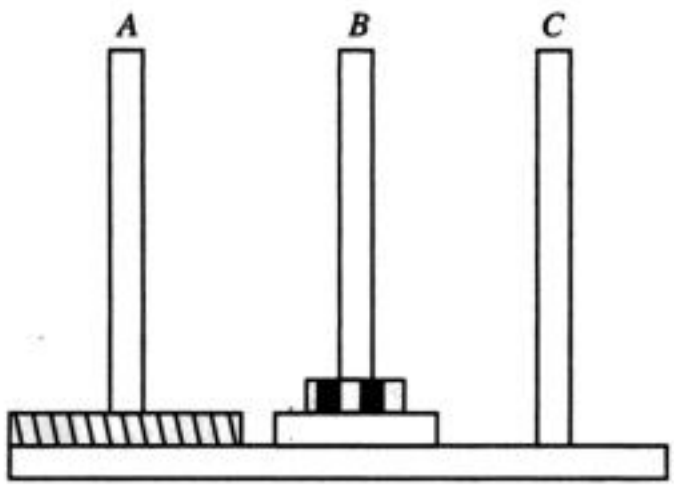


Figure 3.4(c) State (*a b b*).

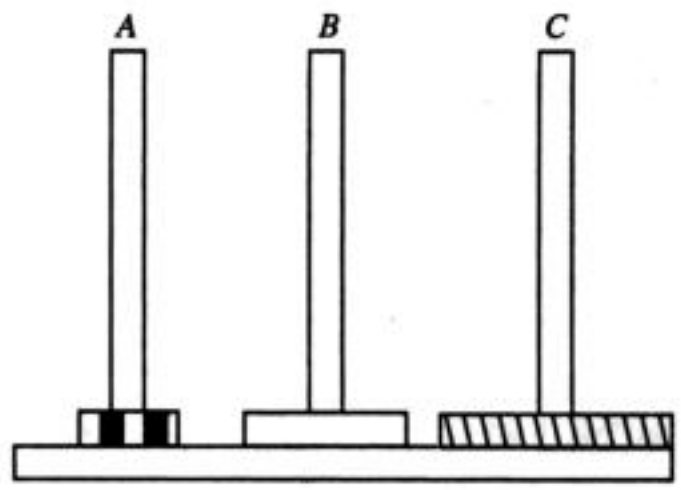


Figure 3.4(d) State (*c b a*).

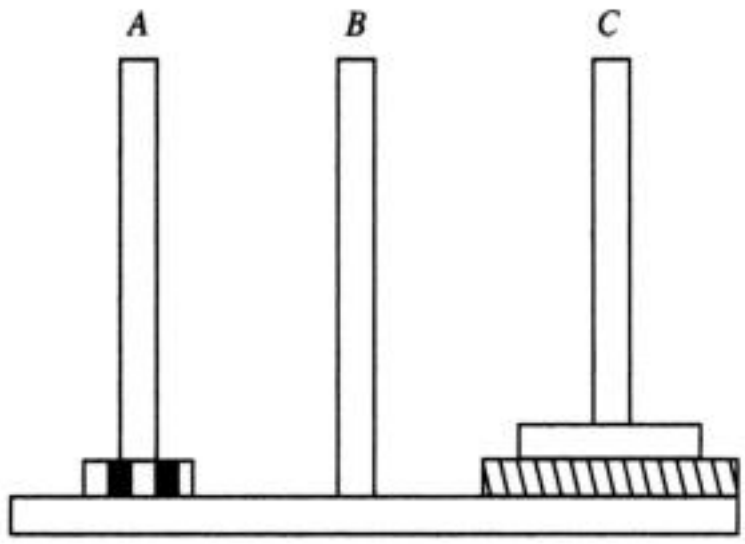


Figure 3.4(e) State (*c c a*).

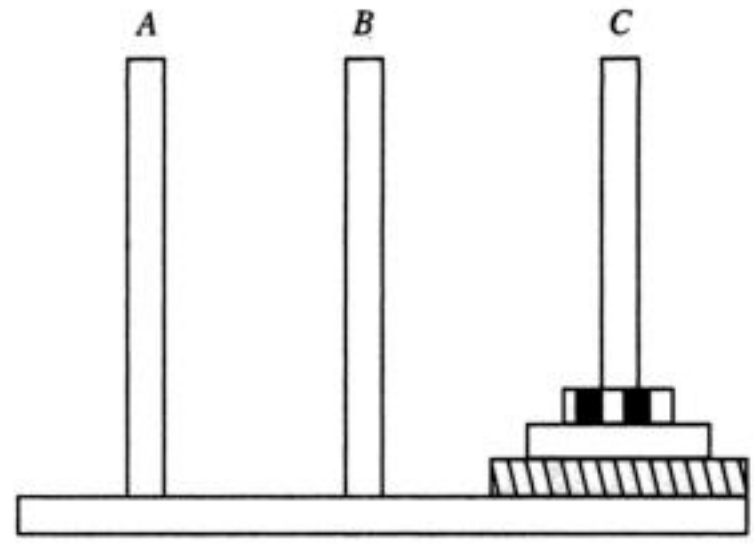


Figure 3.4(f) State (*c c c*).

Monkey Banana Problem

It is used in AI to demonstrate common sense reasoning. In a room there is a monkey, a box and a bunch of banana hanging by some means. The problem is for the monkey to get the banana by the following set of actions.

The actions in sequence are the monkey reaches to the box, pushes the box under hanging banana, climbs on the box and grasps the banana. The state of problem consists of representing the states by four elements (*X*, *W*, *Y*, *K*); where

X = horizontal position of the monkey,
W= monkey's position on the box if yes $w = 1$, not $w = 0$,
Y = horizontal position of the box,
K = grasp of bananas by monkey if $k = 1$ yes and $k = 0$ then no.

Four operators govern the four possible actions. These are:

Go to (*p*):	monkey goes to horizontal position *p*.
Push box (*q*):	monkey pushes the box to horizontal position *q*.
Climb box:	monkey climbs top of the box.
Grasp:	monkey holds the bananas.

p and *q* are the variables to represent any position in the room for the monkey and box, respectively. The actions performed on the particular state results in a variable state as shown below.

STATE DESCRIPTIONS	ACTIONS	RESULTS
(*X*, 0, *Y*, *K*) The monkey and box are at different positions.	go to (*p*)	(*p*, 0, *Y*, *K*) *X* becomes a variable *p*.
(*X*, 0, *X*, *K*) The monkey and box are at the same positions.	push box (*q*)	(*q*, 0, *q*, *K*) monkey and box have the same common position variable *q*.

$(X, 0, X, Z)$ Common static position for monkey and banana.	climb box	$(X, 1, X, K)$ X a common static position when monkey is on box.
$(Z, 1, Z, 0)$ Z is the location on the floor directly under banana.	grasp	$(Z, 1, Z, 1)$ The monkey on the box grasps the banana.

INFORMED SEARCH

Hill Climbing

The hill climbing method is like the depth first search method where the most promising node is selected for expansion. At each point in the search path, a successor node that appears to lead quickly to the top of the hill, i.e. goal is selected for expansion. An evaluation function is required to incorporate task-specific knowledge into the control strategy. The steepest ascent hill climbing or gradient descent search is a useful variation of the simple hill climbing. After selecting the most promising node on the basis of some heuristic evaluating function, no reference to the parent node is retained. The process continues from node to node with previously expanded node being discarded.

With the use of reliable function, which guides the search to a global goal, it has an advantage of substantial savings over uninformed blind search. The search may terminate before reaching the global goal node due to the problems such as foothills, plateau and ridge, etc.

Foothills: It may fall into the local maxima, which is better than some neighbouring state but not better than some other states further away. When local maxima occurs almost within the sight of a solution, it is called foothills.

Plateau: Flat areas of the search space in which a whole set of neighbouring states have the same value. It is very difficult to find by local comparisons, the best direction for move.

Ridge: When several adjoining nodes have higher values than surrounding nodes and have a slope, it is impossible to traverse a ridge by a single move due to the orientation of the high region.

The steps to overcome these problems are to use backtrack to some earlier node and try for going in a different direction for dealing with the local maxima.

For the plateau type of problem, a new selection of search space is found by a big jump.

Moving in several directions at once, by applying two or more rules before doing the test is the best approach.

Breadth First Search (BFS)

As mentioned earlier, the 8-puzzle problem's initial goal and states are given, we apply breadth first and depth first search to get the solution. Campbell (1960) suggested that uninformed or

unguided search problem as blind variation and defective survival process which is fundamental to all inductive achievements and contains the wisdom about the environment is achieved by this process.

In the breadth first search strategy, all nodes at a given level are explored before moving to the next level. One can always find a minimal path length solution when one exists. It uses a generic structure to hold all generated but still unexplored nodes.

The search process starts by expanding the initial state and if none of these nodes are goal node then all the nodes are expanded to the next level by applying all applicable operators to produce all successors of each of the nodes turn by turn. The process continues till the goal node is achieved. The number of nodes grows exponentially from one level to another with increasing depth. The completed time is also exponential and it is useful only when one has an unlimited time and memory. The algorithm for BFS is as follows:

Begin initial state: open

Initialize: = [start]

Closed: = []

- Start with open containing the first initial state until a goal is found or there are no left on OPEN.

  do: remove

- Remove the particular node say *M* from OPEN
- Get goal: if *M* is goal then return SUCCESS
- Generate children of *M*;
  Put *M* on CLOSED

If the children of *M* are already on CLOSED or OPEN, ignore it. Put remaining children on left end of the OPEN.

The above algorithm is implemented on the graph as shown in Figure 3.5, and the result of each step runs into certain iterations as shown in the figure.

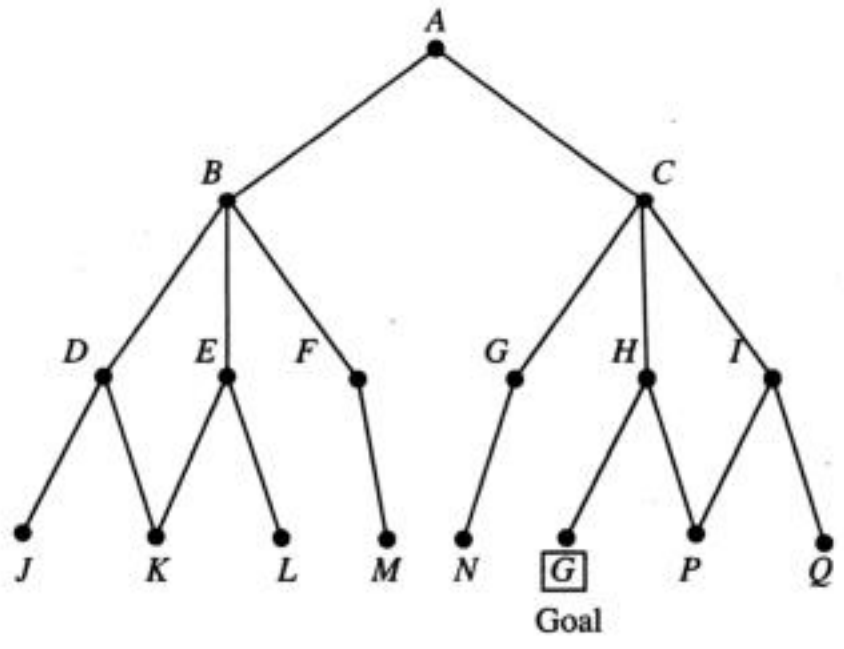


Figure 3.5 Breadth first Search.

Open = [*A*]; Closed = [ ]

Open = [*B*, *C*]; Closed = [*A*]

Open = [*C*, *D*, *E*, *F*]; Closed = [*B*, *A*]

Open = [*D*, *E*, *F*, *G*, *H*, *I*]; Closed = [*C*, *B*, *A*]

Open = [*E*, *F*, *G*, *H*, *I*, *J*, *K*]; Closed = [*D*, *C*, *B*, *A*]

Open = [*F*, *G*, *H*, *I*, *J*, *K*, *L*]; (as K is already open); Closed = [*E*, *D*, *C*, *B*, *A*]

Open = [*G*, *H*, *I*, *J*, *K*, *L*, *M*]; Closed = [*F*, *E*, *D*, *C*, *B*, *A*]

Depth First Search (DFS)

In this search strategy, a random node is selected and search is performed on the descending level until the goal state is achieved as shown in Figure 3.6 which shows the way one proceeds to reach the goal state. Unlike breadth first search (BFS), it requires text memory resources as it considers a limited search space to reach a goal. It is faster than BFS. But if a solution path is located, there is no guarantee that it is the shortest one. Another disadvantage is that if the search of a path is stopped before reaching the end of the path, one can find the solution even if it exits. For illustration an 8-puzzle problem is given below with given start and goal nodes the arrow shows the way it proceeds from start to goal node.

The number in the (·) shows the order in which the nodes are explored for searching the goal state. The depth of the start node is zero and in general the depth of a node equals to one plus the depth of the shallowest parent with the start node.

The complicity of the depth first search tree is $O(b^d)$;
where d = goal depth. Thus, it has exponential time and space complicity. The algorithm for the DFS is given below

- Put start node in the OPEN list.
- If OPEN empty, exit. Else continue.
- From the OPEN, remove first node and put in CLOSED list. Call this n.
- Check if the depth of n equals the depth bound, then go to the second step. Else continue.
- Expand n by generating all of its successors, put them at the beginning of OPEN and provide pointers back to n.
- Exit with the solution obtained by tracking back through the pointers, if any of the successors are goal nodes. Else go to the second step.

As an example in Figure 3.7 a DFS for an 8-puzzle problem is given where the number in circle shows the steps followed in the search process

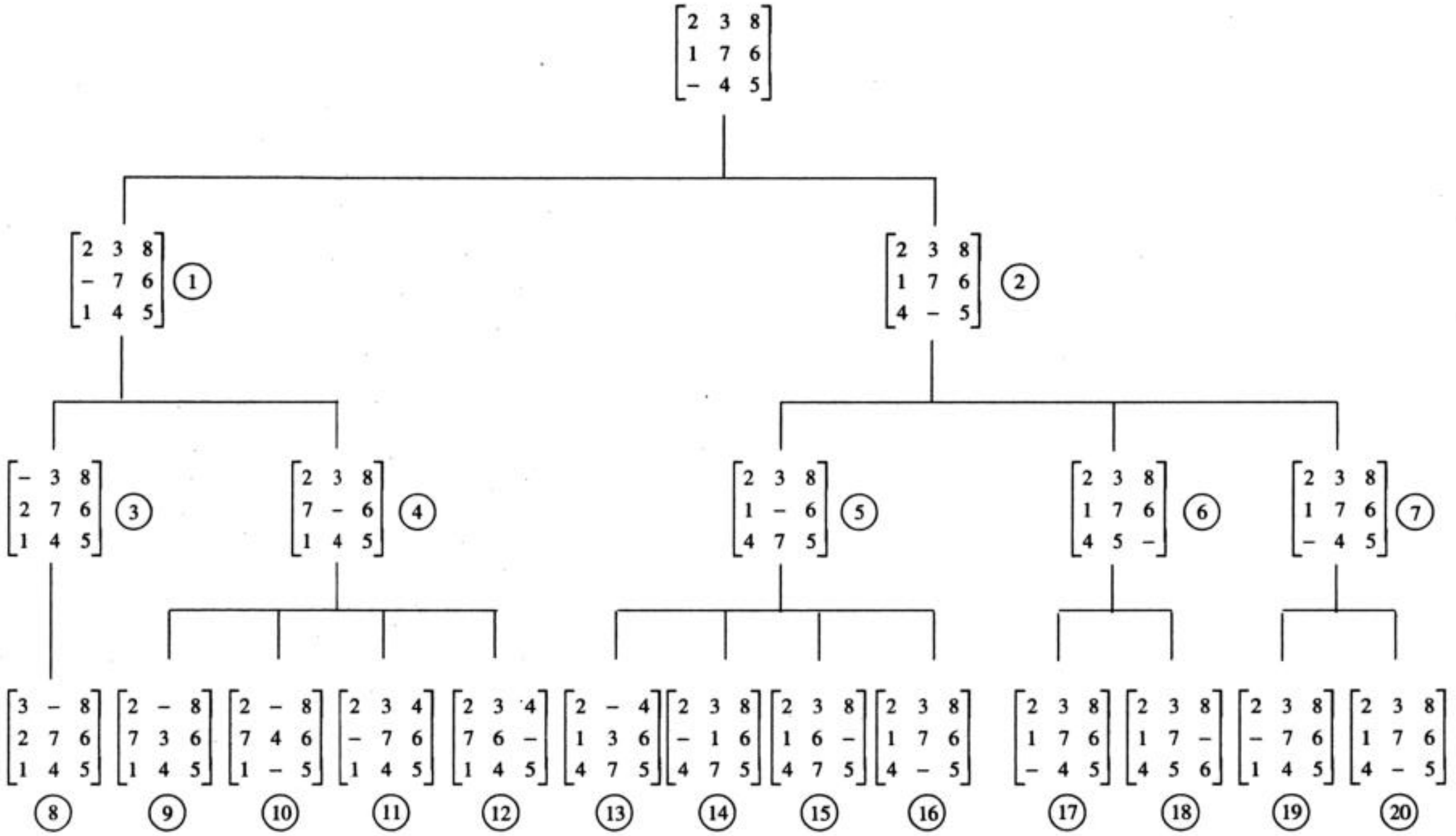


Figure 3.6 Breadth first search.

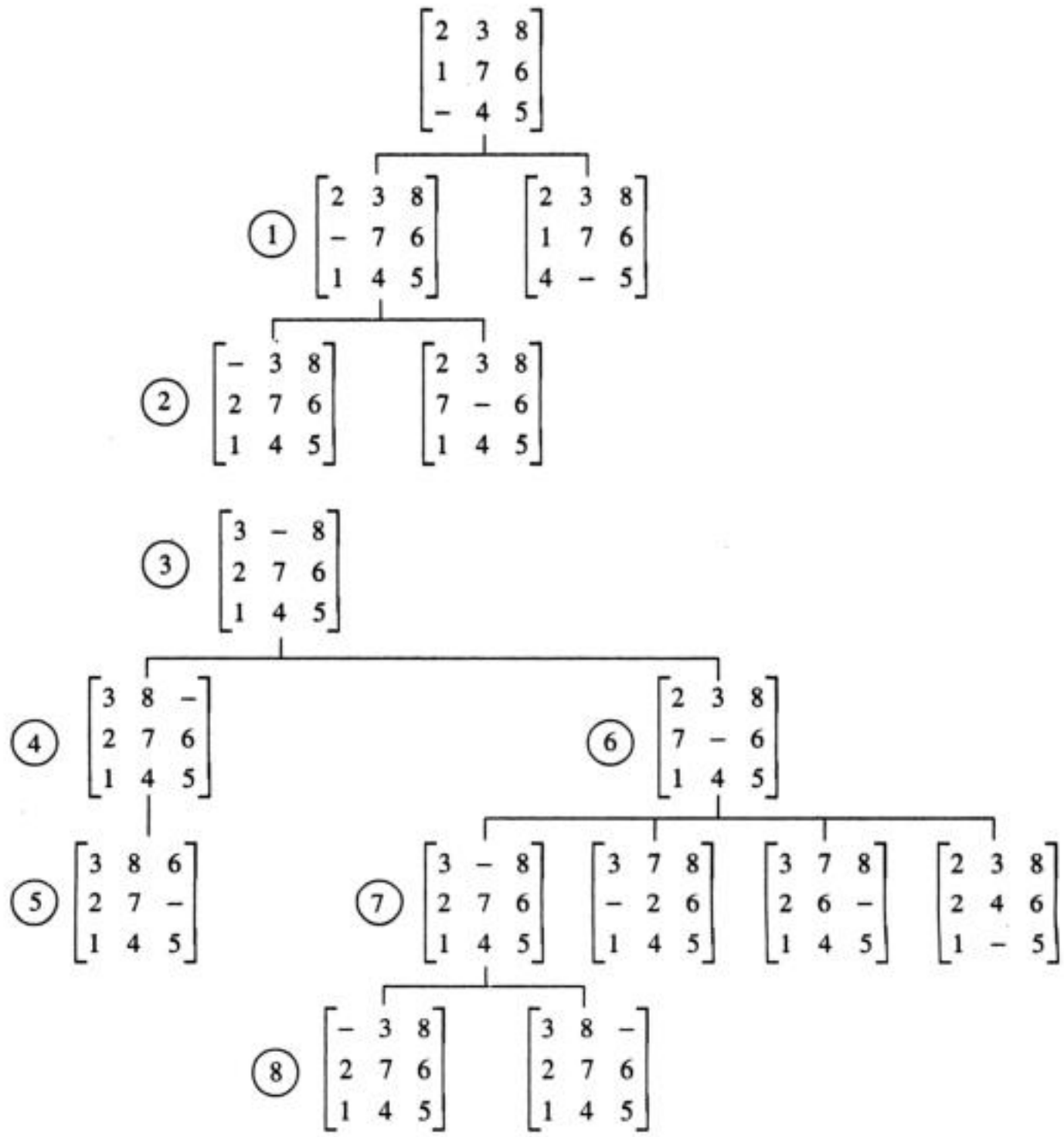


Figure 3.7 Depth first search.

EVALUATING FUNCTIONS AND ORDERED RESEARCH

Evaluation functions are the means to provide a basis for the expectations of a node, which may be the best path from the current to the goal node. The function may be deterministic or probabilistic. The distance or difference metrics between an arbitrary node and goal node is one of the deterministic functions. The node with the lowest value of function is subject to expansion and such procedure is called *ordered search algorithm*. A function $f(n) = g(n) + w(n)$ is defined where $g(n)$ is the length of the path in the search tree from start node to node n. $W(n)$ is the total number of misplaced tiles in the state description problem.

The ordered search algorithm to this puzzle problem is described as follows: Suppose the start node and the goal node are as shown in Figure 3.8.

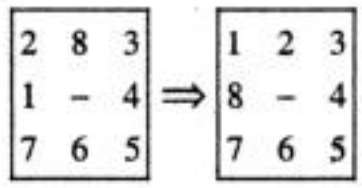


Figure 3.8 States for ordered search.

The $g(n) = 0$ as length of the path from start node to goal node is zero. For $w(n)$ the misplaced tiles corresponding to the goal node are 2, 8 and 1, i.e. the total number is 3 for $w(n)$. Thus, giving $f(n) = 0 + 3 = 3$; the expanded as shown in Figure 3.9. Value of $f(n) = 4(1 + 3)$, $4(1 + 3)$, $6(1 + 5)$ for three-state description consecutively from the left side. Since the two descriptors have the same value, hence both are expanded. And one with the lowest value of $f(n)$ is selected for further expansion. The process continues till one gets the goal node.

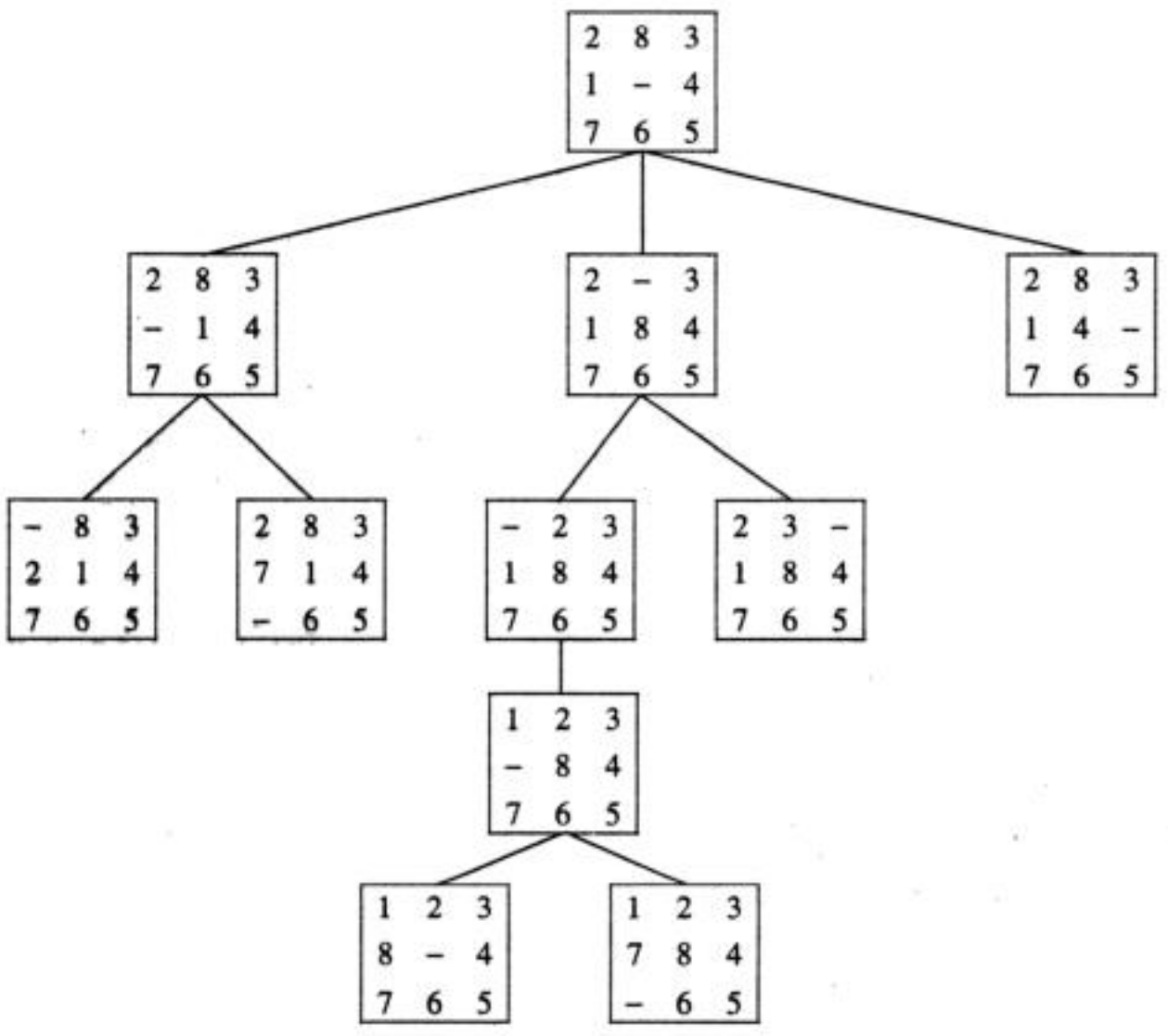


Figure 3.9 Ordered search.

INFORMED SEARCH

The uninformed search methods are best First, A*, and branch and bound which are illustrated as follows.

Best First

In the best first search strategy, at each node of expansion, the path which seems the most promising that can lead to the goal is selected. The selection of promising node is based on some heuristic function. The estimating function, may be, an estimate of the cost of getting to a solution from a given node, is applied to each of the important steps of node expansion, and path selection is shown in Figure 3.10.

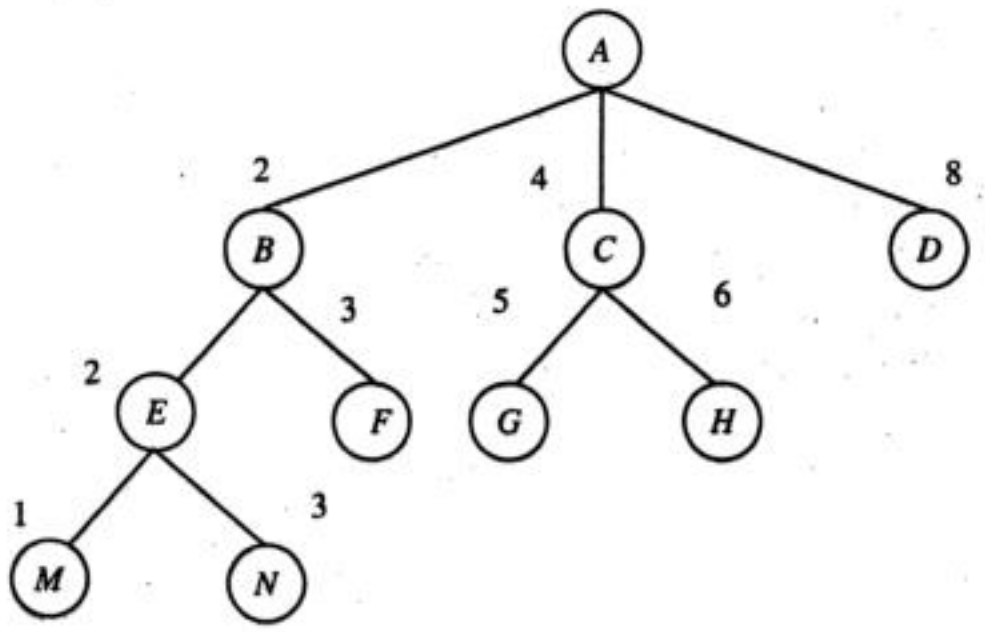


Figure 3.10 Best first.

The node when expanded has the branches $B[2]$, $C[5]$ and $D[8]$; the values in the brackets are the cost of estimating the goal. There can be a number of steps to reach the goal. The best and promising among them is $B[2]$ which is further expanded to have the branches $E[2]$ and $F[3]$. Another path $C[4]$ also looks promising and it is also expanded but when its cost is evaluated as $G[5]$ and $H[6]$, this look less promising. The control is moved to the path through B to E. E is expanded to M and N, and when their cost is estimated it becomes $M[1]$ and $N[3]$. The node $N[1]$ is only one step behind the goal node. It is why the path $A\ B\ E\ M$ is selected, the steps involved are as follows:

Step 1: Put the initial node in OPEN.

Step 2: Pick the best node on OPEN, use heuristic method to find the promising node, generate its successor.
Follow the steps to each of its successor; do the step 2 till OPEN is not empty and the goal is not achieved.

Step 3: Evaluate the non-generated nodes so far, add them to the list of OPEN and record their parents. If they have been generated before, change the parent if new path is better than the previous one.

A* Search Algorithm

Defining the following parameters for Figure 3.11, the algorithm performs the implementation of graph A.

OPEN: A list of known next node to be considered.

CLOSED: It contains all those nodes already encountered and incorporated in the output structure.

K: The current node under consideration.

L: It is the expansion of nodes in *K* less than those in OPEN or CLOSED.

The input graph had been described by Webber and Nilsson (1980) and its corresponding output graph is shown in Figure 3.9.

The following steps are performed to implement the search strategy.

Step 1: Put the first node in OPEN as OPEN (1).

Step 2: Remove it from OPEN, place in both CLOSED and *K*.

Step 3: Expand the node to *L*(2, 3). Since there are no other nodes visited at their initial stage, this expansion is placed directly into OPEN (2, 3).

Step 4: The nodes in OPEN are sorted according to some control strategy or by defining some heuristic function or cost function for each link or branch of the expanded node. For this problem new nodes are put on the tail of the list.

Step 5: Reduce OPEN (2, 3) by removing the head [2], the head is added to CLOSED (1, 2) and placed in *K*(2), expand this node (4, 5). Any node in OPEN (3) and CLOSED (1, 2) are removed before placing them in *L*.

Step 6: Add *L*(4, 5) to the tail of OPEN (3), such that OPEN (3, 4, 5) becomes the new value and the node (2) is expanded to the branches (4, 5).

Step 7: The sequence in OPEN is sorted, the process is repeated by removing the head (3) of the list OPEN (3, 4, 5), adding it to CLOSED (1, 2, 3) and placing it in *N*(3).

Step 8: Continue the previous steps till the goal node is achieved.

A* Algorithm

The control strategy for guiding the search requires an evaluation function. A function f is defined so that the value $f(n)$ at any node n is the actual cost of any optimal path from nodes to any node plus the cost of an optimal path from node n to a goal node. That is,

$$F(n) = g(n) + h(n)$$

If $h(n) = 0$ for all times, i.e. cost estimation to the goal, the search will always find the optimal solution if it exists. The search when $h(n) = 0$ becomes breadth first search in which no cost is involved from node n to goal node. $h(n)$ large value and do decreasing it by one for each level of exploration. Making $g(n) = 0$ in such situation lead to depth first strategy.

A trade-off between the two extremes $h(n) = 0$ and $g(n) = 0$ is required. Suppose $h^*(N)$ is the actual cost of start to current node n, then $h(N) \leq h^*(N)$



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

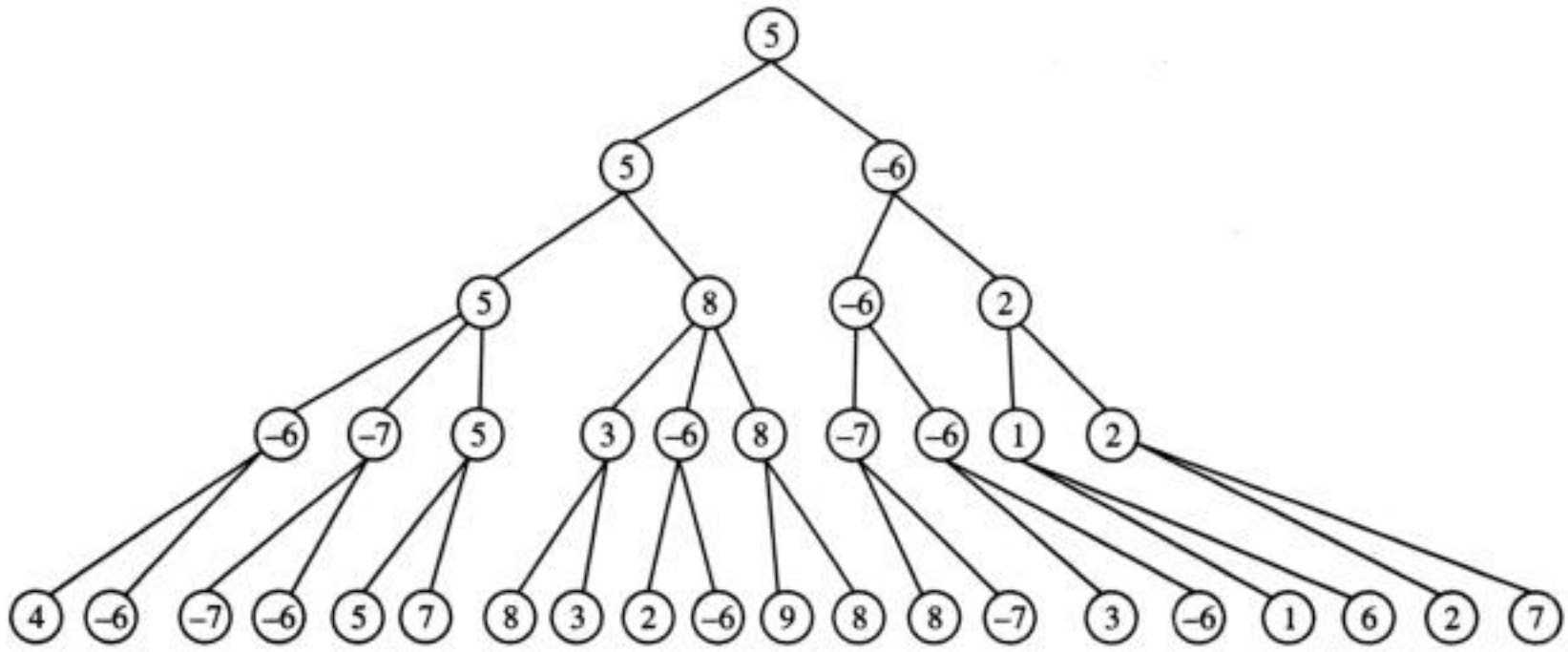


Figure 4.5 Game tree with the values of interior nodes filled in.

It is clear from Figure 4.5 that moving upwards to the parent node from the leaf nodes, the parent nodes occupy the minimum of the leave nodes, i.e. 6(4, –6), –7(–7, –6) and so on. Similarly, the next upper layer has the value of its nodes as 5(–6, –7, 5), 8(3, –6, 8), –6(–7, –6) and 2(1, 2), i.e. the greatest value in the nodes as its outcome if first player moves the minimum of value 5(5, 8) and –6(–6, 2) are outcome of second player is moves and finally the root node has 5(5, –6) the maximum of the first player's move. Thus, the first player wins the game by five units. The minimax search of the game tree to depth d examines b^d leaves in which every position has b legal moves.

ALPHA-BETA SEARCH

At the ply Z the leaf nodes are 5, 6, 7, but the leftmost node at this ply requires min (6, 5, 7) equal to 5. The leaf nodes of middle node have values 2 and 8, respectively. The first leaf below Y is 2 and it has min 2. Since root node will not accept it because it takes 5, so it will not thrice Y. Therefore, other successors of Y are not considered. Here the α–β pruning is deployed. These leaf nodes of Y containing value 9 and 8 so these nodes are pruned.

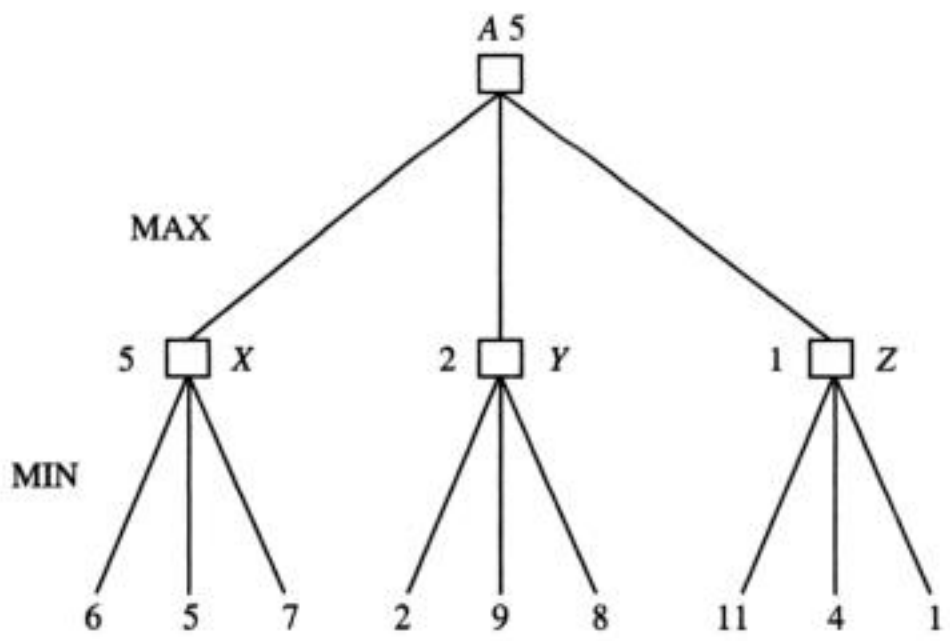


Figure 4.6 α–β search.

The first leaf below Z has the value 11 so Z is worth at most 11. This is greater than the at most value 5 at root node. It imposes a bound on the max value of root node as 11. The second successor of Z is 4, so exploring continues to the next node, which is 1. The node Z becomes 1. Finally, the root node has the choice to select the maximum among the $X(5)$, $Y(2)$ $Z(1)$ which is 5.

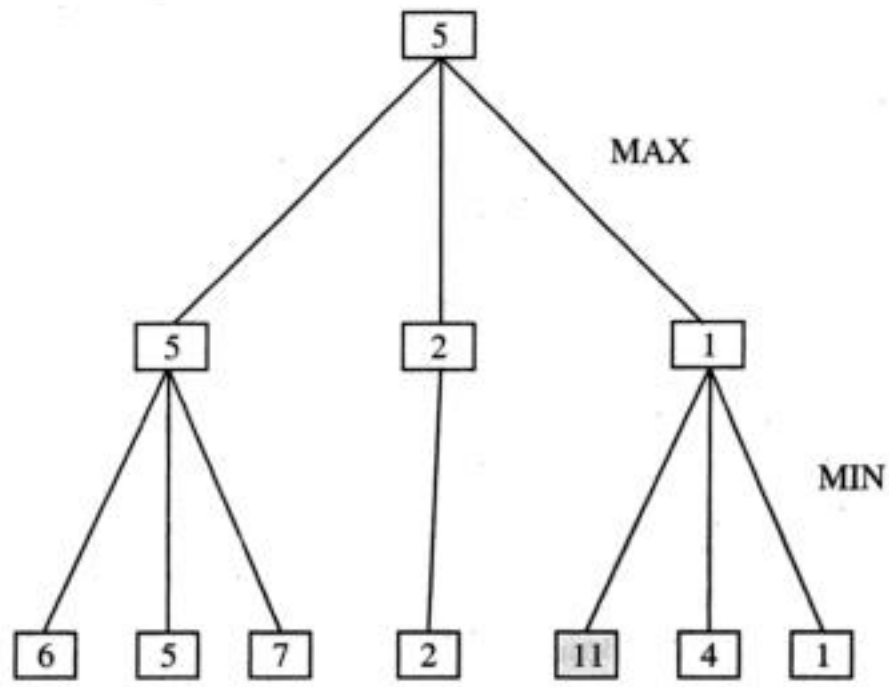


Figure 4.7 α–β pruning.

Iterative Deepening

The iterative deepening algorithm was originally used for the program called CHESS# 4.5 (Slate and Atkin, 1977). It is an iterative process of search in which at each iteration, the search moves to the next deeper level by discarding the nodes searched in the previous iteration or level. It means it starts with depth search to a depth of one ply. And after applying its static evaluation function to the results of each of its node, it then starts search over depth of second ply. If goal is not achieved, then it discards all nodes generated and moves to the next ply. The iteration mechanism of search is shown in Figure 4.8.

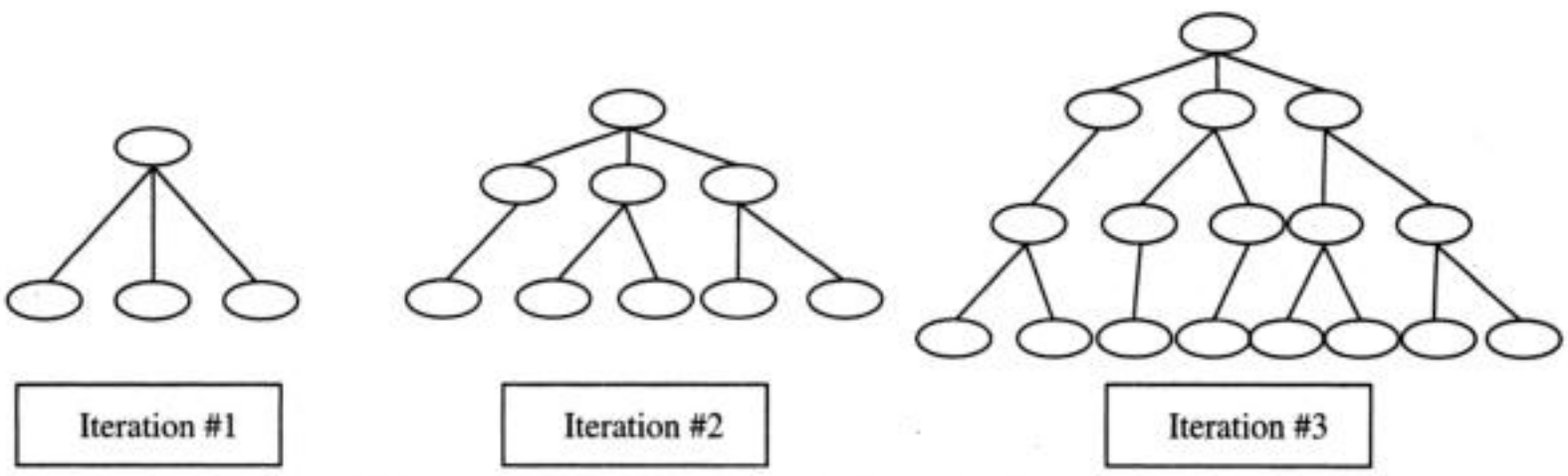


Figure 4.8 Iteration steps in Iterative Deepening.

The iteration continues till a new node has been found. The Iterative Deepening is useful in game playing, because most of the game playing programs are time constrained. For instance, one may require to finish the game in a given time, but it may not be possible but with the Iterative Deepening, the current search can be stopped, at any time and the best move found

by the previous iteration can be played. The alpha–beta procedure can print many more branches with effective ordering and total search time can be decreased drastically, resulting in more time for further iterations. A combination of depth first and breadth search was also developed naming depth-first iterative deepening (DFID).

The demerit of this algorithm is that it performs warded computations before reaching a goal depth.

It is optimal over depth first and breadth first search methods; the time and space complexities of this search are $0(bd)$ and $0(d)$, respectively.

The basic algorithm steps are:

- Initialize the iterative depth search: *ids* = 1.
- Perform the depth first search to a depth of *ids*. If a solution path or goal is achieved, then return it.
- Else, do *ids* = *ids* + 1 and go to the previous step.

The performance of A* may be improved by using iterative deepening, as its large memory requirement is reduced considerably. This is called IDA* algorithm which is widely used in robotics path planning and other games of large search origin.

PUZZLE SOLVING

There are three well-known sliding tile puzzle problems: 3 * 3 eight puzzles, 4 * 4 fifteen puzzles, and 5 * 5 twenty-four puzzles. The fifteen puzzles invented by Sam Lloyd in the 1870s is the best known so far. Computational complexities of each of the puzzle problems are enumerated with the corresponding evaluation and search algorithms.

There are 9!/2 (181, 440) states in the 8-puzzle problem. Its complexity in its move has been enumerated by Schofield such as average length of shortest distance between two states is about 22 moves and the maximum distance between any pair of state is 22 moves.

There are $16!/2 \approx 10^{13}$ states in the fifteen puzzle which can exhaust the computer memory in minutes. The 5 * 5 puzzle takes $25!/2 \approx 10^{25}$ states, which requires impractical time.

Algorithm

The A* algorithm is the widely used algorithm which guarantees to find the optimal solution by introducing a heuristic evaluation function. The heuristic function in these problems is Manhattan distance and is computed by counting the number of grid units that each tile is from its goal position and summing these values of all tiles. A* is not suitable for fifteen puzzle, as it does not provide optimal solution, because every generated state is stored in the memory. Based on the depth search strategy, a more efficient algorithm, i.e. Iterative Deepening A* (IDA*) which reduced considerably the memory requirement and also generated the solution near A*. The same Manhattan distance was used as the evaluating function. The Manhattan distance is a lower bound on the optimal solution length. For the 24-puzzle problem, an admissible evaluation function was linear conflict enhancement to the Manhattan distance

(Hansson et al., 1992). It states that if the tiles are in their current row, but reversed relative to their position, then at least one tile must temporarily move to the row, to allow the other to pass, and then move back. Assuming that there are no interactions between the tiles, the Manhattan distance is the sum of the distance each individual tile has to move considering each pair of tiles and computing the move of each tile and interactions between pair of tiles, i.e. pairwise distance which equals the Manhattan distances. The average solution length was about 53 moves for the problem, which generated billions of nodes pairwise distance. The solution length was obtained by more 100 moves for some problem generated trillions of nodes. It is near to conflicts and pairwise enhancement with IDA*.

Rubik Cube

In the early 1980, Erno Rubik from Hungary invented Rubik Cube, producing the same impact on masses as puzzle problem. In this problem, general problem solver strategy is implemented, by generating and ordering subgoal. Macro operators, which are composed of sequences of primitive as individual twists the cube. Automatic learning of the macro operator is programmed to solve legally and scrambled 3 * 3 * 3 cube with average solution length about 86 moves. The IDA* algorithm with heuristic based on pattern databases is utilized to obtain optimal solutions. Herbert Kociemba of Germany developed similar ideas. The standard Rubik's Cube consisted 27 cubbies; the 20 movable cubbies include eight cubbies on the cube's corner and 12 cubes on the edges. The total number of different permutations and orientations of the corner cube is 88, 179, and 840. Using breadth first search, the exact number of moves required to solve each state of the corner cubbies and store their values in memory can be computed. It requires zero to element moves, each entry with four bits, for a total of 42 Mbytes of storage. Another problem is to find how many moves about two states can be used for the shortest solution path. It is 20 moves for $3 \times 3 \times 3$ and 11 moves for 2 * 2 * 2 Rubik's cube.

Michael Buro from NEC Research (Buro, 1999) has described some of the methods such as GLEM, Multi-Prob cut and opening book algorithm which are gain independent. The basic procedure and objective of these algorithms are as follows.

- The development of a hardware-based algorithm that makes a trade-off between computational strategies of the algorithm with its hardware implementation in such a way that allows game free searches even with poor evaluation function.
- Determining an efficient and smart, i.e. intelligent evaluation function which calls for adaptation or self-tuning according to the complexities of the problem solution space in particular and nature of the game in general.
- Performs selective searches according to the variations in one another's move at different level or state of the game. Irrelevant moves are to be avoided which may no longer affect the decision process or the goal in the long-run.
- The different phases of the game including standard should be updated using opening book bargaining strategy.

There are some important landmarks in the development of game playing algorithms which are given below.

The research by IBM in game playing program was Deep Blue. Deep Blue was developed by C.J. Jan, Murray Campbell, Feng Hsiung, Joe Hoane, Jerry Brody and US chess champion Joel Benjamin (Iyengar et al. 2000) which combined brute force search and sophisticated knowledge of chess. The brute force search, sometimes used in a derogatory context as search itself is a dynamic knowledge. It uses large deep searches to compensate for inadequate knowledge. The idea culminated in the Deep Blue victory in an exhibition match with world chess champion Garry Kasporov in 1997. Deep Blue used a 32 node IBM Rs/6000 SP high performance computer chat uses power two super chip processors (P2SC). Each node produces 256 processing that work in tandem. It searched 200 million chess position per second. Deep blue considers material, king safety, board piston and tempo (a measure related to the aggressiveness of a player's strategy relative to the way the game unfolds).

IBM's Deep computing institute aims at deriving and integrating concepts pertaining to search strategies such as pattern recognition knowledge discovery bioinformatics, computation biology, optimization and data mining and neural net methodology.

GLEM

In this approach, the two phases are built up for the construction of evaluation function. One is the selection of features, which corresponds to the few distinct numerical properties of the position, and another is the rule of combination of these features. In the game-playing program, selection of features is the most important step, with the simple reason that it requires a domain specific knowledge as well as the programming expertise. As already implemented by Arthur Samuel chess playing game, weighing factors assigned mainly to the features, for the given training position is not a difficult task as compared to the selection of features itself. The automatic evaluation function construction in the GLEM was developed by the efficient procedures for getting training position, exploring the feature space and fitting feature weight (Buro, 1998). In this approach, a large number of features that are Boolean combinations of atomic correlations, are linearly combined to construct evaluation functions, rather than the nonlinear combination of a few features. In the Logistello game, 100,000 binary features in conjunction with 1.2 million automatically tuned parameters were generated instead of a few manually selected features.

An expert knowledge-based program was developed, based on this approach which could beat the best players in human beings. The KBS consists of a set of encoded 8 million configuration weights.

This approach has been applied to other similar games such as GO. In this game, brut force search is incompetent because of a very large branching factor. It requires an improved evaluation function.

MPC (Multi-Prob Cut)

The minimax and its efficiency improved approach alpha–beta algorithm requires a great amount of search time to search the entire game tree up to a certain depth. The suggested search procedure is selective searches, which permits pruning of sub-trees that are unlikely to



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

For this reason, the programmer can make only declarative statements and ignore the underlying implementation.

Inference process

It has the following steps for its implementation in PROLOG.

- PROLOG starts its search from the top of the database. When a match occurs, it returns the goal; for example, if a question is asked as such as surface (X) in the example is given below. The control will search for a predicate surface (.) in the database from top as well as some instantiation of X that will make the statement surface (X) true. An uninstantiated variable will match any argument that happens to be in its position, in fact. For example, in the question X = smooth if user enters semicolon after getting the goal, then X = rough as the next goal. Since the previous goal has already been reached, it would try to match the next similar fact with uninstantiated argument X. After the goal is achieved X = rough.

Examples

1. Size (small)
2. Size (medium)
3. Size (large)
4. Taste (sweet)
5. Taste (sour)
6. Smooth (x)
7. Rough (x)
8. Cost (mango, low)
9. Cost (apple, high)
10. Buy (Y, X):- sweet (X), has (Y, money), cost (X, low)
11. Sale (Y, X):- sour (X), Large (X), rough (X)
12. Cost (Y, X):- small (X), sweet (X)
13. Preserve (Y, X):- smooth (Surface), cost (X, high)
14. Exchange (Y, X):- smooth (Surface), cost (X, Low)

- When PROLOG finds a match and instantiates the appropriate variable, it leaves a *printer* where the current match occurs.
- When a goal matches, the head of rule rather than a fact along with the body of the rule is treated as subgoals that must be satisfied to prove the goal, i.e. head of the rule.

Suppose ? – Exchange (*A*, *B*)

The question matches with head of the rule 14 in the above example. The variables *A* and *B* are instantiated with another set of variables and then it becomes exchange (Y, X). Then the control takes the first literal from the body of the rule and would try to match with any of the rule. It matches with rule 6, i.e. smooth (X) becomes smooth (surface) by instantiation of X=surface.

Then the next literal cost (X, Low) is taken for consideration.

Programing in PROLOG

Programing in PROLOG is easier than programing in procedural and functional languages provided in the problem solution where the logic dominates over mathematics. In this section, PROLOG programing components or constituents are outlined. Then through examples of a small program and programs for some of the problems are described in other chapters.

Programing preliminaries

The program structure of PROLOG looks like:

```
Domain      : Domain
Predicates  : Goal
              Database
Clauses     : Predicate
              Clauses (rules)
```

Variables: A variable is free uninstantiated if it does not have value at a particular time. It is called bound or instantiated if it has a value at a time. Once the goal succeeds and the variable bindings are displayed, it frees all variables again and backtracks trying to make the success of goal again.

The clause eats (Som, _) states that Som eats everything. Everything is represented by underscore. If one writes goal: eats (Som, Mango), the goal will succeed through its matching with eats (Som, _).

Rules: The rules are expressed as facts with IF and THEN. The comma (,) and semicolon (;) stand for AND and OR relationship. For OR relationship, two separate rules are preferred.

Variable in Rules

```
Electrical:-  parts (Machine, Stator)
              & parts (Machine, Rotor)
              & parts (Machine, Brush)
```

The stator rotor and brush are the variables of an electrical machine, which are expressed in rules. It states that if it has its parts as stator, rotor and brush then the machine is electrical.

TRACE: The execution of a program in step at a time can be obtained through 'trace', added anywhere before the domain section.

UNIFICATION: Unification is a pattern matching process. It has the same concept as mentioned in Chapter 2.

Predicates unify with each other, and it matches with rule and cost (mango, low). Thus, the goal is achieved through the matching of the literal of the rule 14.

PROLOG contains three basic control strategy:

1. FC matching (Unification)
2. Backward (Backtracking)
3. Execution of the query
4. Control Commands in PROLOG

PROLOG provides a system defined predicate called 'cut' (denoted by !) for affecting the procedural behaviour of program and to limit non-determinism by preventing interpreter from finding alternative solutions.

Cut prunes the search tree and, hence, shortens the path traversed by PROLOG interpreter which reduces the computation time and also saves storage space. To illustrate 'cut' more clearly, let us consider the following rules:

```
G     : -P,Q,R, ! ,X,Y,Z      (1)
G     : -R,S,T.               (2)
Goal  : ?-G
```

PROLOG system backtracks among subgoals P, Q, R until R succeeds. Then it causes 'cut' to be crossed to the right to reach subgoal X. Backtracking can occur among X, Y and Z to satisfy the entire conjunction of sub goals.

- If G succeeds to produce solution using rule (1), then rule (2) is not tried for alternative solution.
- If G fails before crossing 'cut', then rule (2) is tried for alternative solution.
- If X fails, then no attempt is made to re-satisfy Z and the entire conjunction of subgoals fails and, subsequently, the goal G fails and rule (2) is not tried.

Red cut: The cut whose removal from the program changes the meaning of the program.

Fail predicate: Predicate fails to tell PROLOG interpreter to fail a particular goal and subsequently forces backtracking. All the subgoals defied after fail will never be executed. Hence, predicate 'fail' should always be used as the last subgoal in a rule.

Cut and fail combination: If cut is used in conjunction with fail predicate and if control reaches fail in the body of a rule after crossing cut (!), then no solution is displayed. Consider the following definitions of the rules using cut and fail combination.

```
X     :- K, !, fail.    (1)
X     :- P, Q.          (2)

Goal :- ?-X.
```

- If K succeeds, then rule (1) fails and rule (2) will not be tried.
- If K fails, then rule (2) will be tried on backtracking.

Recursion in PROLOG

Recursion relation or mathematical definition for computing factorial of positive integer is defined as:

$$Factorial(n)\begin{cases}1, & if\ n=0\\ n*Factorial*(n-1), & otherwise\end{cases}$$

In PROLOG, program for computing factorial of a positive number is written as follows:

```
% factorial(N,R)–succeeds if R is unified with the factorial of N.
factorial (0,1).                                                    (1)
factorial(N,R): – N1 is N–1, factorial (N1, R1), R is n*R1.         (2)
```

Rule (1) states that factorial of 0 is 1 which is a terminating clause. Rule (1) is a recursive clause which states that factorial of N is calculated by multiplying N with factorial of $(N - 1)$.

Program design steps

In the development or design of a program, the overall structure contains compiler directives, domains, global domain databases, predicates, global predicates, and goal clauses. Not necessarily all the components of a program as mentioned above should be present. But the essential components are domains, database predicates, goals, and clauses. The domain pertains to the variables, i.e. variable is equated to the type of variable, i.e. symbol, string, character as mentioned earlier. For example, in a question-answer session regarding the parts and functions of a machine are written as:

```
Domain
Parts, function  = Symbol
Query            = String
Reply            = Character
```

Database: The answer to this query is either Y (positive) or N (negative). The database is represented by

```
Database  Y positive (functions)
          X negative (functions)
```

Predicate: Predicates are the facts, which are represented in standard form of predicate as mentioned in the section. For the problem of classification of different types of machine such as electrical, mechanical and electronics depending on the component parts and functions, the facts are expressed in predicate form as:

```
Hypothesis (Machine)
Function (Function)
Response (Reply)
Go
Positive (Query, function)
Clear facts
Remember facts (function, reply)
Ask (Query, function, reply)
```

The predicates are user's model of representing the facts. The facts in the rule are represented in the predicate form as mentioned above. If a fact is represented like Resistance_correlates (voltage, current) in the rules, then it must be shown under the heading of predicates. The clear facts are in-built predicates to make screen clear with the facts.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

purposes and aims of the object even if their attributes and services are the same, these can't be associated with single class. But if the purpose is to pronounce these as a possession, then single class can be associated with them.

Relationship

The relationship or the association between two objects, either of the same class or of different classes or of two classes may be binary or *n*-ary. The association among the classes may be one to one, one to many, many to one and many to many as shown in Figure 5.7.

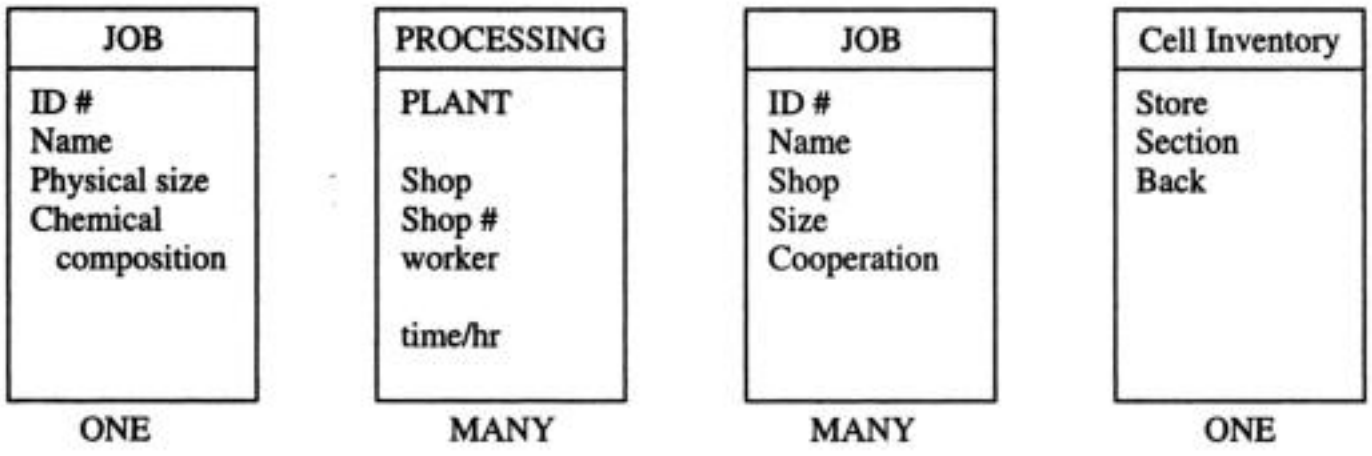


Figure 5.7 Relationships in OOM.

Properties and Features of OOM

Encapsulation

Encapsulation is the basic property of an object, which encapsulates the data, and information it contains. An object contains the data and function which can be performed by the object, not by any outside agency. The object also covers a shield around it, which is called interface that permits access to the other agency through certain services. The operations, performed by the object inside, do not form the part of interface commands or services. For example, a CPU has its resister and ALU to do certain functions but other devices as outside object that can have certain services to send (in) data and receive (out) data through some interface (port). CPU as an object encapsulates its internal function and has interface to outside objects. Similarly ROLL list is an object, which has add-name and change name as interface and attendance as internal function.

- Encapsulation is like a bodyguard and abstraction to VIP. It provides an explicit boundary between an object abstract interface (its abstraction) and its internal implementation details.
- It tells the users what to do and what not to do.
- The integrity of data is performed because the access to the encapsulation of data is limited to the operations defined on the data.

Polymorphism

Polymorphism is the property of an object which deals with the concept that an object can be of different types at different times. It is also the capability of different objects to respond to



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

Hierarchical system in Figure 5.8, represents an industrial management of human resources. The superclass is managing director whose attributes are qualification and experience, which inherit all of its class from one level to another level of the hierarchy. There are three management levels in the industries as given topmost (level 1) managing director, followed by a group of general managers for different activities: production, administration and marketing at level 2. In the next level 3, plant manager and chief engineer are the two subclass of GM 1, GM 2; Administration has two subclasses personnel officer and estate officer. Sales manager and material resource manager are the two subclasses of GM 3, marketing.

Hierarchy types

There are two types of inheritance. One is strict in which a subclass takes all the properties and also functions as inheritance of the parent class. And the other is non-strict inheritance when some of the features of the parent class are available in the subclass. For example, in the subclass at level 2, the executive engineer has the derived feature as control and management from the superclass GM 1 of level 2, but the design attribute is specific to it, an example of non-strict inheritance. In the level 2, all the properties of MD of level 1 are derived by the GM 2: controls, management, planning, etc. and also another attribute, maintenance is added which pertains to specific to the plant, equipment and structure.

Multiple inheritances

When subclass or derived class is inherited through many or multiple classes or base class, it is called multiple inheritances. For example, in the hierarchical model of the library users is shown in Figure 5.9. The users are teachers and students.

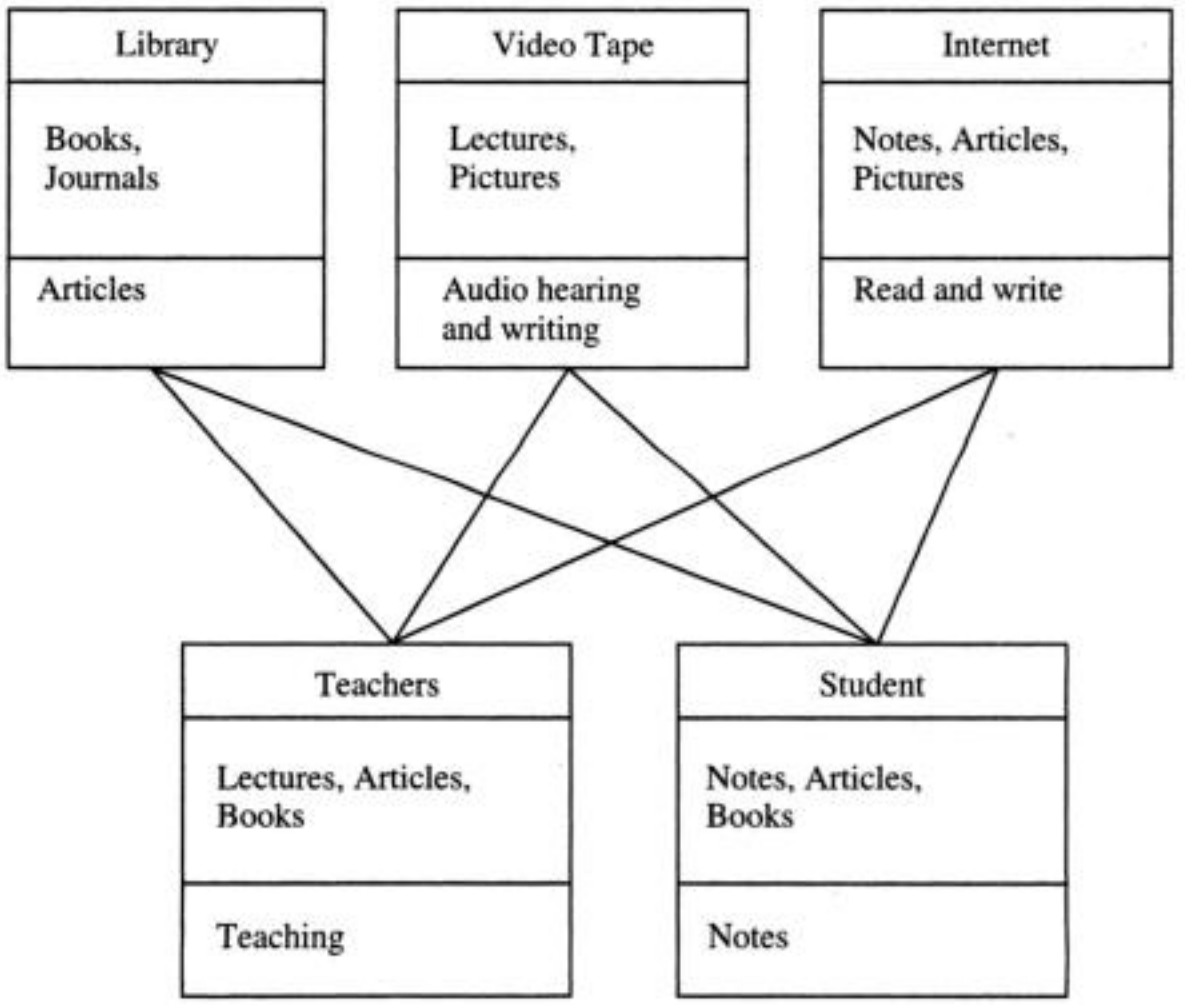


Figure 5.9 Multiple inheritances.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

- Abstraction of instance variables and instance methods public and private.
- Encapsulations of variables private and of methods public and private.
- Unit is the kind of modulus for modularity it supports.
- Multiple hierarchical inheritances it contains.
- Strongly typed and polymorphism.
- Concurrency is exhibited through multitasking.
- It has no persistent objects.

SUMMARY

LISP, PROLOG, and object oriented programming (OOP) are the languages that are extensively used in AI-based methods.

- LISP is a functional language which does not distinguish between data and control. It is list processing mechanism.
- The list, input, output and list manipulation commands have been described with suitable examples.
- PROLOG is a programming logic-based language. The syntax, semantics, predicates, clauses rules with examples have been discussed.
- The salient features of OOM and C++ with the features of the AI-based OOP have also been described.

EXERCISES

5.1 Write the return value of the following

(a) car '(b c)
cdr '(a b c d e)
cons ''a' (p q)
list ('c'(a b))
member 'd' (a b d)

(b) setq normal-temp 92.3
setq cm-per-mm 10
setq pi 3.14
(add-em normal-temp cm-mm pi)
(setq x' B)
(list 'A X'C)

5.2 What do these following functions represent

(a) (IF(<test>)(<then>)(<else>))

(b) DFUN FACTORIAL(n)
TIMES n (FACTORIAL (-n 1))); recursive call

(c) <Parameter symbol> <initial value> (update expression)

5.3 (a) Write a program using DO function to transpose a row into vector.

(b) Write a function in LISP, find the sum of two rows, each with three elements

4 8 12
7 14 21

5.4 Write the return value of the functions

(a) (/(+(*63)&)2)

MIN (783)

(b) NOT (AND (NIL NIL))

EQ (B B)

MAP CAR-2 #' + (8 2 4)' (5 1 3)

5.5 Write a program using list function

(a) To find minimum and minimum number in a given list

(b) To access a value in a given array

(c) To find union of two sets.

5.6 Write a program to implement breath first and best first search methods.

5.7 Describe the salient features of LISP with comparison to PROLOG and OOM. Mention some of the limitations of LISP and different versions of LISP with their salient features and limitations.

5.8 Develop a hierarchical inheritance in OOM for the following problems with attributes of each class:

(a) University-Faculty-Department-Centre-Laboratory

(b) Chief-Superintendent-Executive-Assistant-Junior engineer

(c) Factory-Plant-Shop

(d) Materials Manager-Regional Manager-Sales Manager-Chief Manager-Sales Manager

(e) State-District-Municipal-Block-Colony.

5.9 Make a multiple inheritance model

(a) Family: income resources

(b) Class room: connected entities

(c) Factory: input resource: product.

5.10 Write a program in C++, PROLOG and LISP for

(a) Inventory control

(b) Time table

(c) Reservation for surface and air travel.

5.11 Make a comparative view of the similar and dissimilar functions, properties and characteristics of PROLOG, LISP and OOM.

5.12 Show the relationship of OOM with AI with illustration to the problem solving related to analysis, synthesis, design in engineering.

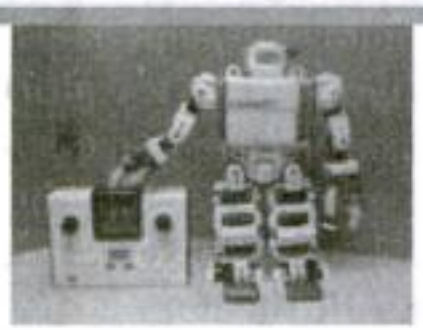


UNIT II

Knowledge-Based Systems

Copyrighted material

Chapter 6



Knowledge Representation

There are various informal methods for knowledge representation (KR), such as Rule-based model (Production system), semantic nets, frame, conceptual dependency, conceptual graph, scripts, OOM and attribute-value pair.

Rule-based model or rule-based system (RBS) is one of the most popular and versatile knowledge representation systems used in different fields. It is easy to design, develop and implement on low cost PCs. The method of KR is based on logic in general and Horn clause in particular, which is IF–THEN structure. It is a logic programming-based structure, which is implemented in PROLOG, which is a programming language. In this chapter, an RBS structure and its various properties, i.e. rule recency, contradiction, reduction, specify, etc. have been discussed. Some of the expert systems, which deploy rule-based system for their KR, are also outlined.

The rule-based system is also known as production system and it was used by Simon and Newell (1972) in modelling the human problem-solving behaviour.

STRUCTURE OF AN RBS

The rule-based system consists of three main components (see Figure 6.1): 1. Knowledge-base; 2. Working memory; and 3. Inference engine.

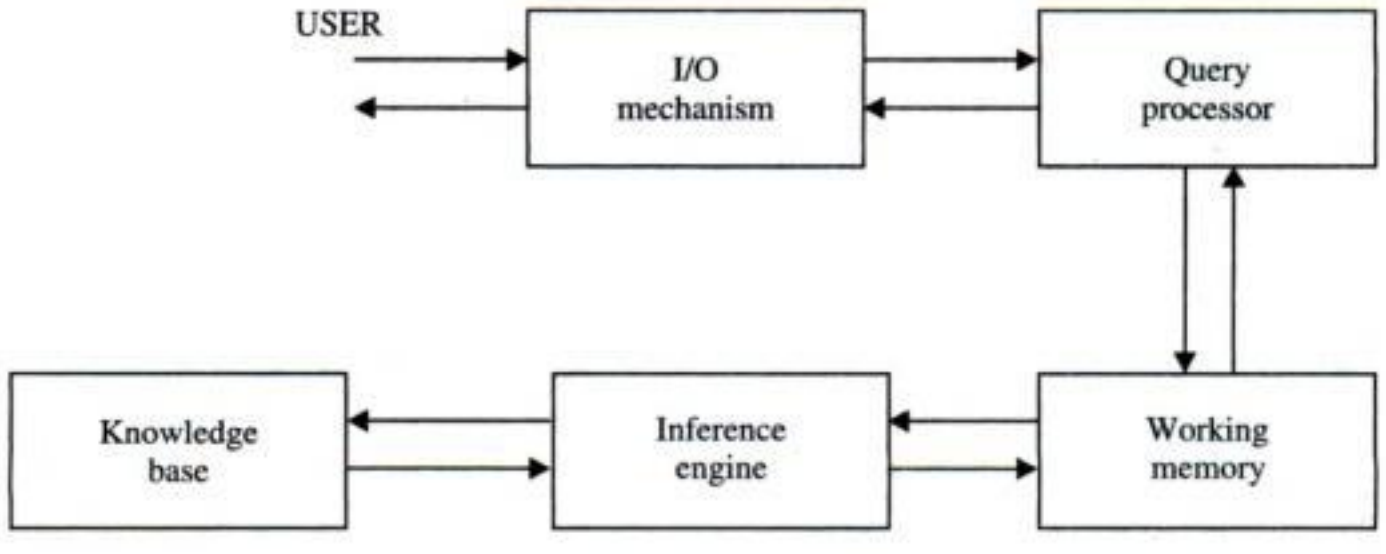


Figure 6.1 Rule-based system structure.

Copyrighted material



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

In the semantic network, there are two types of nodes (i) type nodes that define classes of objects, and (ii) token nodes, which are the specific instances of a type of object. For instance, an animal or a mammal would be example of the type node while Lola would be a token, or a particular instance of the type mammal.

Salient features

Semantic distance: The closeness of the concepts is measured by the number of nodes between the 'concept nodes'. This measure was obtained by Collins and Quillian (1972) on the reaction-time studies of the associative networks. The lesser the number of nodes between the two concepts nodes, the concepts in semantic, the more will be they closer to each other.

Multiple inheritances: A node may have inherited properties from its different links coming from its upper level, which is called *multiple inheritances*. The multiple inheritances give rise to conflict where there is contrary information from different links.

Scooping: This deals with the problem of defining the scope of a quantified variable. If a statement S is true for values of a variable V, then it is said that S is scoope of V Partitioned nets as mentioned above implement this.

Demerits

In general the set of nodes connected to a super node should be independent of each other so far as the syntactic or semantic relations are concerned. In certain situations, this may not be true. For example, in the classification problems such as birds (sparrow, parrot, peacock, pet, wild), the nodes originating from birds are not independent; say, parrot be pet, peacock be wild. In such situations, inadequacy of semantic net is found when there is interelationship among the classified objects.

Applications: Winston (1970) used semantic nets in structural descriptions in a program called ARCH. Carbonell and Collins (1970) used semantic net to define South America in geography in his program SCHULAR.

Phillip's semantic net

Katz and Fodor (1963) proposed "The structure of a semantic theory" in which they brought two main aspects of natural language processing. One thing made distribution between semantic and syntactic structure more widened and another was the mechanism by which the semantics of a sentence could be analyzed in terms of semantic features. Quillian in 1966 made an attempt to implement a program based on the concept of Katz and Fodor. TLC (Teachable language comprehender) builds a connective network called semantic network from the semantic memory when a set of facts for simple sentence was inserted into it.

Phillips (1978) justified the semantic structure in terms of linguistics, psychology and sociology.

His primitives were VAR (variety) which is a category/subcategory relationship between nodes, IST (INSTanu) is category/element relationship between the nodes. TYP (TYPically)

and MAN (MANifestation) link a category node with some activity that is typical or some alternative views of a category.

From Figure 6.9, it is clear that the primitive VAR is connecting the different subcategories of four-wheeler and two-wheeler. "Transports are manufactured" is a manifestation link, the category 'Transport' to the some other view of the 'transport'. Car is a VAR of four wheelers and 'Bedford' is an aliment connected to the category. 'Car is a cabin type' is a typical structure of a car and further 'car is a cab' and 'car is racing' are the two activity based or alternative views.

Bedford is manufactured, can be inferred from the property inheritance in the structure. Since Bedford is a car and car is a four-wheeler and four-wheelers are transport and the transports are manufactured, as it is being manifested, therefore, Bedford is manufactured. Philips's primitives achieve greater discriminations than Quillian's primitives.

FRAMES

Minsky (1985) developed the frame model of representation of knowledge primarily in some other contexts such as storage of different events in a mind as a frame of reference. When a particular event or situation arises, all the information, which is stored in the memory of mind, is referenced as a particular frame. The frame contains the particular objects and the associated properties or attributes of the object corresponding to the particular situation. For example, driving a car, the event is concerned with the object car which has the properties of a type of car, its attributes as gesture and posture, engine, speed, colour, shape, cost, etc. and the value of the attributes as average speed 100 km/h, blue colour, low cost etc. There are mainly three aspects: one is object, its attributes and values. The object is termed frame, the attributes as slots and values are the default value or expectations for the slots. Similarly, for a computing machine as a frame, the slots and corresponding default values are shown in Figure 6.11. The general property of the frame is that a frame may contain a frame.

General frame structure is shown in Figure 6.10 (Despain and Patterson, 1978) and the corresponding example for a building is, as frame and its parts as slots with field are values. A frame has different slots and a slot has different facts.

{<frame name> { <slot1> {<fact1> <value1> .... value n1>}}}

{ <fact2> <value2) ..... <value n2>}

{ <slot2> {<fact 1> <value1> .... value n >}}

.

.

{<slot m> {<fact m1> <value1> ..... value n_k}}

Figure 6.10 A general frame structure.

The general structure may look like this way also.

(Frame (slot (fact (values)))), i.e. a frame

Building {<House> {<Room> {<Bedroom> <Drawing room> <Bath room>}}}

{<Doors> {<Main> <Side> <Back>}}

{<College> { <Room> { <Classroom> <Laboratory> <Library>}}}

< {<Shopping> { <G-floor> { <Counter> <Hall> <goods corner>}}}

A building as a frame, can be an object having different kinds such as house, college, and shopping complex and with different slots, room and doors as two facts for house with corresponding values of room are as bedroom, drawing room and bathrooms. Similarly, college and shopping complex are given with corresponding fact and values.

Hierarchical view

A top level view of a frame system is shown in Figure 6.11, a computer can be represented with three links: a kind of AKO, a part of and instance of computer is a kind of machine; parallel computer is a kind of computer; vector register is a part of supercomputer and Cray2 is an instance of parallel computer.

Basically, a frame is looked upon as a form to be filled. The structural frame system with objects and links can be shown as given below. Cray is the instance of parallel computer.

Type: value concept.
Memory: value string.
Input device: value string: min 1: max 5.
Output device: value string: min 1: max 3.
Processor: value: min 1: max 16.
Define an instance of Cray, with valid input of all the parameters.
Define: Cray instance of parallel.
Type: value, instance.
Memory: GB.
Processor: value 16.
Input: value keyboard.
Output device: CRT.

Figure 6.11 Hierarchical view of a frame system.

Properties: There are various properties associated with a frame, regarding their structure, values and procedures.

Nested structure: In programming, the nested loops are loop within a loop, similarly, a frame within a frame or a slot may contain a frame. In the above example, the building is considered as a frame, the room may be taken as one of the frames. It means above slot filled by lower level frames. A frame is composed of the interrelated frames that are required to represent a domain.

Values: The values may be constrained. For example, the height of a man cannot be greater than 18 ft., weight not more than 300 kg, exceptions are exempted to be quoted here. So the constraint should be imposed on the values. Also the range of values should be specified in terms of minimum(min) and maximum(max) as mentioned above. The min and max show the number of items the fact should be in the range of value which can be shown with predicates such range (M, N).

Default value: When a frame is called upon by default, the slot and the associated values also come into view. For example, if a frame 'Building' is considered then rooms, doors and size, etc. are by default get associated with the frame.

Procedure in frame: The slots are associated with the procedural information. The most common examples of such procedures being attached to the slots are *if needed* and *if added.* If needed procedure describes the process required to establish a value for the associated slot when a value is required, if added procedures are event-driven processes that are triggered when the associated slot has been assigned a value.

CONCEPTUAL GRAPHS

Conceptual graph (CG): A conceptual graphs a system of logic based on Pierce's 'existential graphs'. CG has a full power of FOL and CG represents higher order logic. Conceptual graph has a finite bipartial graph. The ends of nodes are concepts and conceptual relations. Conceptual relations have one or more arcs, each of which is related to some concepts.

Canonical graphs: The canonical graphs are the distinguished meaningful graphs that represent real or possible situations in external world. Formation rules are the basis for derivation of new canonical graphs from other canonical graphs. There are five formation rules for deriving a conceptual graph *W* from conceptual graph *U* and *V* are organized into hierarchy such as horse is sub type of Arabian and Arabian is a sub type of mammal.

Marker specific but unnamed individuals are indicated by CG. Marker is a unique token written as a number preceded by #. Like horse, #248 indicates a horse having number 248 as the marker.

Referent: Each concept node can indicate an individual of a specified type, which is the referent to the concept being individual or generic. For example, in figure, the concept [PERSON: Sam] has the type person and the referent Sam. Sam and wine are individual concepts since they refer to specific entities whereas DRINK AND JAR has no referent fields since they are generic concepts; which refer to unspecified entities.

Copy(U): It produces a duplicate copy of a CG. *W* is the exact copy of *U* or *V*.

Restrict(U, C, L): It modifies the graph by replacing a type label of a concept with the subtype or a specialization from generic to individual by inserting a referent of the same concept. For example, one can replace animal with a horse.

In the formal way, any concept C in U, type(C) may be replaced by a subtype T, if C is a generic. Its referent may be changed to an individual marks I where $L = T{:}I$. These changes are permitted only if referent(C) conforms type(C) before and after the change; i.e. $T::I$. The use of restriction rules is to make two concepts match together so that join can be performed. **Simplify (U, R, S):** if conceptual relations in the graph U are duplicates, then one of them may be deleted from U together with all its arcs.

Join: It combines two graphs into a single graph. If a concept $C1$ in U is identical to a concept $C2$ in U; then join (U, $C1$, $C2$) to obtain the graph by deleting D and linking to C all arcs of conceptual relations that had been linked to D.

Fuse: For two disjoint conceptual graph U and V, if a concept C in U is identical to a concept D in V, then fuse (U, V, C, D) is the graph obtained by deleting D and linking to C all arcs of conceptual relation that had been linked to D.

CG to FOPL

Changing conceptual graph to the first order predicate logic is given by Sowa (1984) in the following manner:

- Assign a unique variable name to each of the generic concept of the graph.
- The DRINK and WATER in Figure 6.12, are assigned the variable name x and y, respectively.
- All types of labels of a concept node such as PERSON and WATER are converted to unary predicates with some name and whose argument is the variable or constant given by that node.

  Represent each N-ary conceptual relation in graph as N-ary predicate whose name is the same as the conceptual relations such as AGENT, OBJECT and INSTRUCTION are converted to predicates with as many arguments as these are connected to the relation. **Concept referents such as Sam and mineral become FOPL constants**.
- Concepts with extended referents such as? map the universal quantifier?

Take the conjunction of all atomic sentences formed by conceptual node and conceptual relation.

- In FOPL, which is the body of the predicate calculus expressions, all the variables in the expression are existentially quantified.

For example: "The every elephant has a trunk" can be represented in conceptual graph as

[ELEPHANT: V] (ORGAN) [Trunk]

The equivalent FOPL representation is given by

$\forall$ x y (ELEPHANT (x)) (Trunk) (y) & ORGAN (x, y)]

Restriction of Two Graphs

Fusion

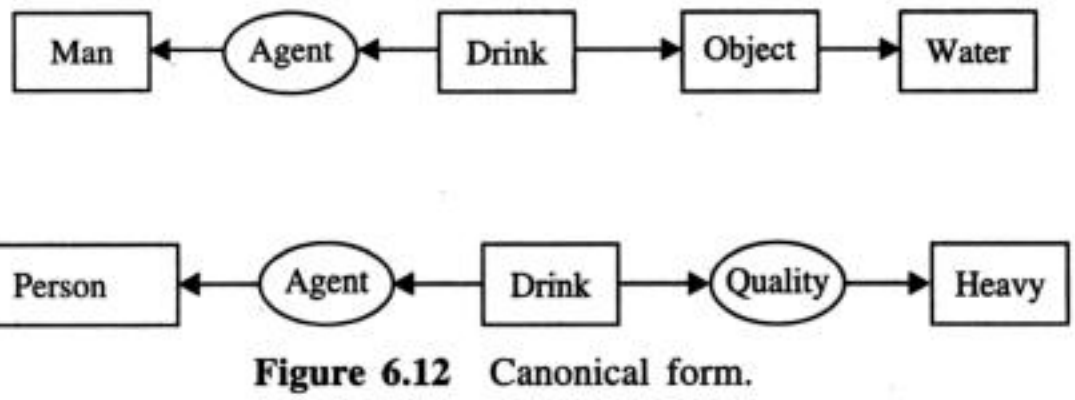


Figure 6.12 Canonical form.

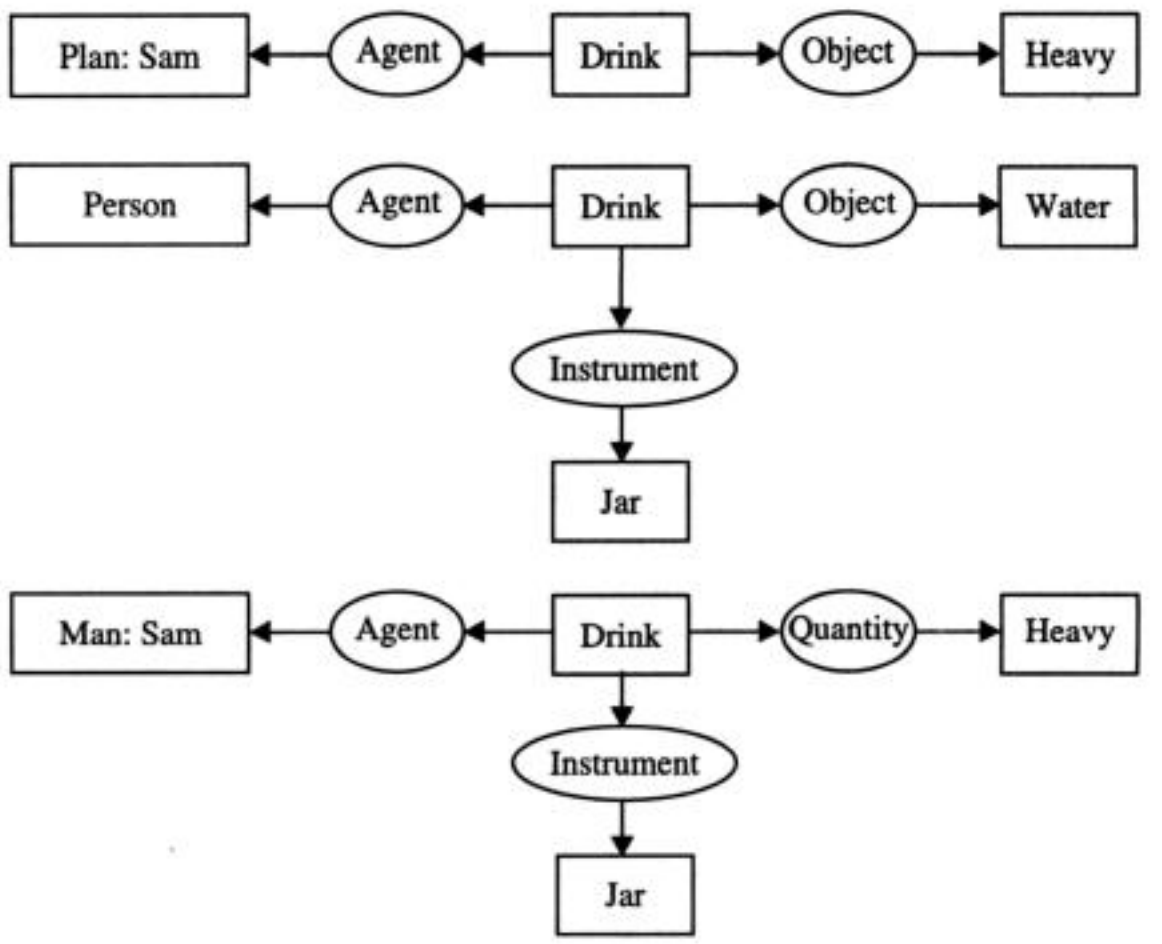
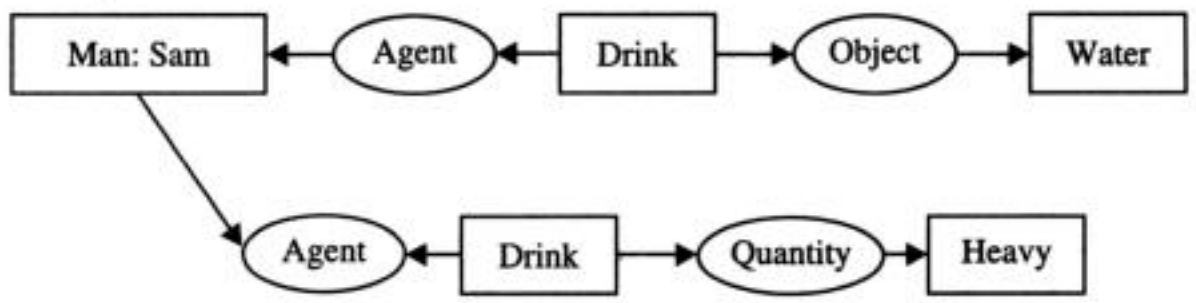


Figure 6.13 Fusion of graphs.

Joining of two Act



Fusion

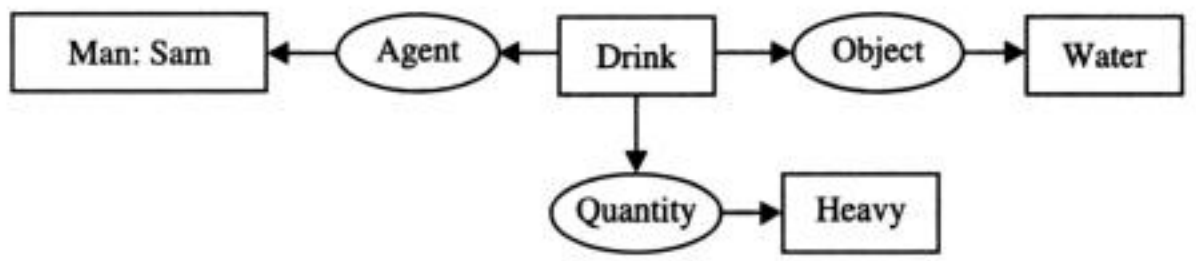


Figure 6.14 Simplification by canonical rules.

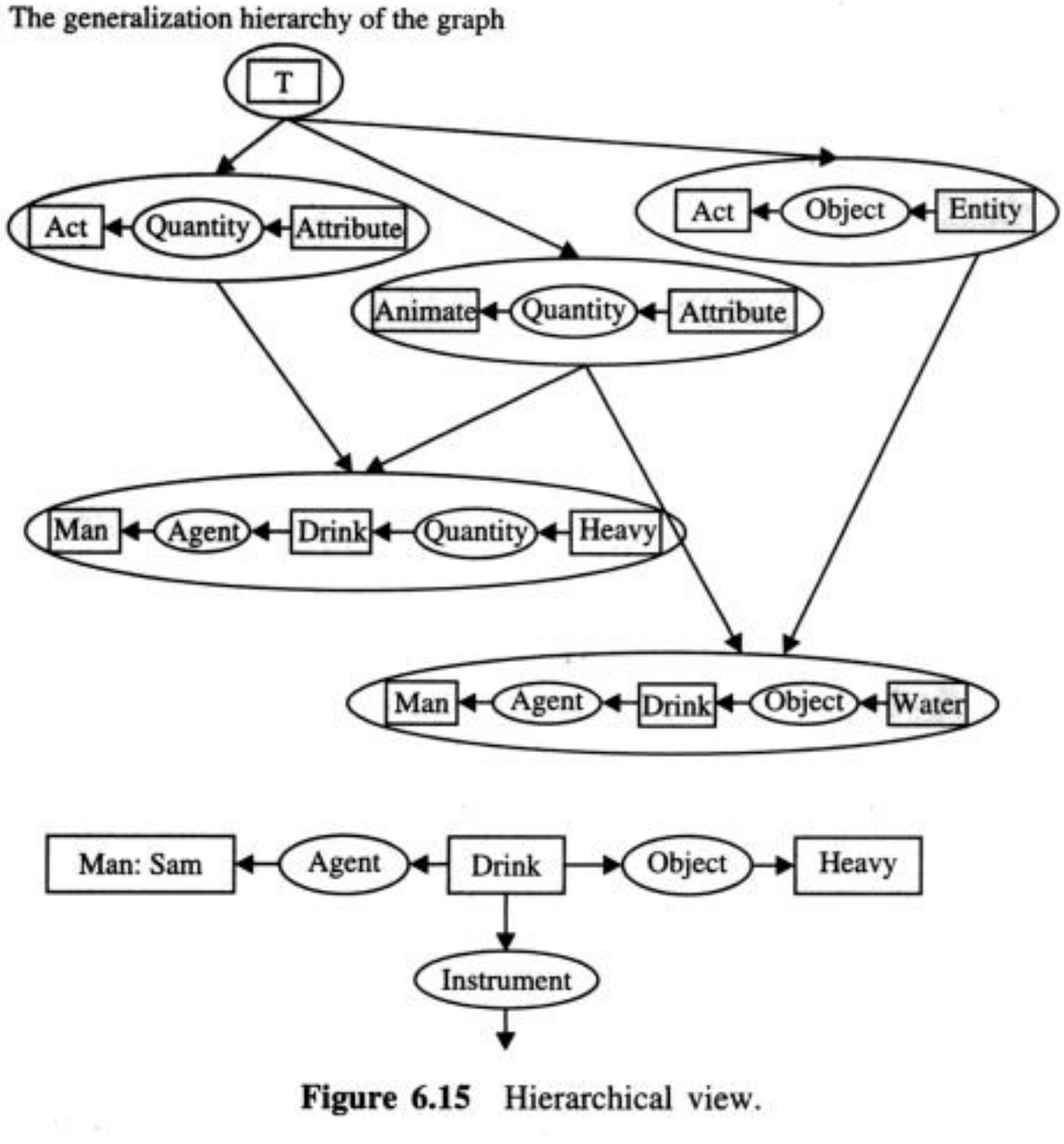


Figure 6.15 Hierarchical view.

Sowa (1984) developed the representation model of the knowledge being perceived by a human being. The perception consists of different concepts and their relationship about physical entity and an abstract or combination of both.

CONCEPTUAL DEPENDENCY

The conceptual dependency is a model to represent knowledge in a natural language sentences. There are certain conceptual primitives that can be combined on the basis of certain set of rules to represent the concept or meaning of a particular sentence. Schank and Riesbeck (1973) presented the theory and further developed by Abelson for understanding the meaning of a sentence. There are various components as the building blocks of such representation scheme, such as (i) entities (ii) primitives action (PA) (iii) primitives conceptual categories (iv) rules for semantic relations or conceptual dependencies and (v) conceptual time and space.

Primitive Acts (PAs)

There are various primitive acts which represent a particular action. The representation of certain actions is also built from these acts. There exists slight differences among the different sets of primitive acts. Following are the sets of PAs proposed by Schank and Abelson (1977).

'ATRANS': TRANSFER OF AN ABSTRACT

(I) Entities: There are four primitive conceptual categoies from which dependency structure can be built. These are also called entities.

ACTs : Actions
PPs : Picture Producers
AAs : Action Aiders
PAs : Picture Aiders. These are modifiers of aiders.

Primitives actions	*Meaning concerned*
ATRANS	Transfer of an abstract relationship.
PTRANS	Transfer of the physical location of an object.
PROPEL	Application of physical force to an object.
MOVE	Movement of a body part by an actor.
GRASP	Grasping of an object by an animal.
INGEST	Ingestion of an object by an animal.
EXPEL	Expulsion of something from the body of an animal.
M TRANS	Transfer of mental information.
MBUILD	Building new information out of old.
SPEAK	Production of sounds.
ATTEND	Focussing of a sense organ toward a stimulus.

Conceptual cases: All actions involve one or more combination of these four conceptual cases which are objective, directive, instrumental and recipient.

Semantic rules: The relationship between an actor and an event, or a primitive action and an instrument, two PPs, etc. are formed by a set of rules. These rules and the corresponding conceptual dependency structures are described below:

R1: It shows the relationship between an actor and the event performed by the actor. It is bi-directional. The K indicates the past tense.
R2: It shows the relationship between a PP and a PA that is being asserted. The attribute of PP (Hari) is PA completion (fair).
R3: It is the relationship between two PPs, each belongs, to a different set and defined by each other like one set of PPs is (Hari, Jack, Janni) and another set of PPs (Doctor, Engineer, Professor).
R4: It describes the relationship between a PP and an attribute that has already been predicted by it. The direction of the arrow toward the PP is being described.
R5: It shows the relationship between two PPs one of which provides any of the three kinds of most commonly used information provider such as possession (Poss-by), Location (LOC), and physical containment (CONT).
R6: It shows the relationship between an ACT and the PP; that is the object of that ACT. The direction of the arrow is toward the ACT.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

9. ACT ← D → PP, Sham ⇔ PTRANS, PP; O: Fertilizer; D → field, bag — Sham fertilized the field.

10. PP <≡ PA, PA — Flower <≡ Dimension > y, Dimension = y — The flower blooms.

11. (a) ⇔ (b) ⇔ Pati ⇔ Propel ← Arrow ← health(–10); R → Shami, bow — Pati hit Shami.

12. ⇔ Yesterday ⇔ John ⇔ PTRAN — Sham went yesterday.

13. ⇔ College ⇔ Rat ⇔ PTRANS ← I ← college I saw I rat. PP — While coming

14. ⇓ Street I ⇔ MTRANS ← O donkey ← I heard a donkey; ⇓ ⇔ ⇓ ⇔ MTRANS ← donkey — CP — eyes on the street CP ears

Other properties

There are four categories of the 12 acts. These are physical with 5, instrumental with 2, mental with 3 and global with 2 acts.

The physical acts are: PROPEL and MOVE

INGEST: Taking something from one's inside. The object of the INGEST must be smaller than the mouth of the actor or must be divided into pieces smaller than the mouth opening, is the food object.

FXPEL: Taking something from one's inside and force it out. Its object must be previously INGESTED.

GRASP: To grasp object which must be within size limit.

MTRANS: It represents a change in the mental control of conceptualizations and underlies verbs like recall, commit to memory, perceive, sense, and communicate.

The donor and recipient are not always two different mental process.

Conscious processor (CP): It is the storage place of the currently being processed entity.

Long-term memory (LTM): It is a passive element whose level of activity is too low to characterize. It is a processor in the sense that forgetfulness and subconscious activities are associated, together with one's belief of the world.

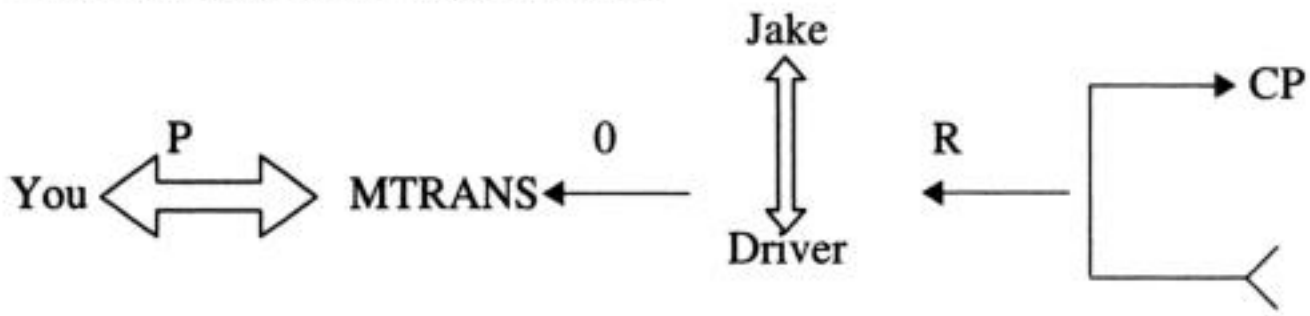
Immediate memory (IM): It is the storage area for the current context being processed as well as the processing area for conscious thought.

Sense organs (eye, ear, nose, tongue, and skin): These are in all preprocess which convert raw information or data or signal into the conceptualizations describing that data.

Body: The internal sensations such as pairs, unease, and excitement, etc. pertain to body.

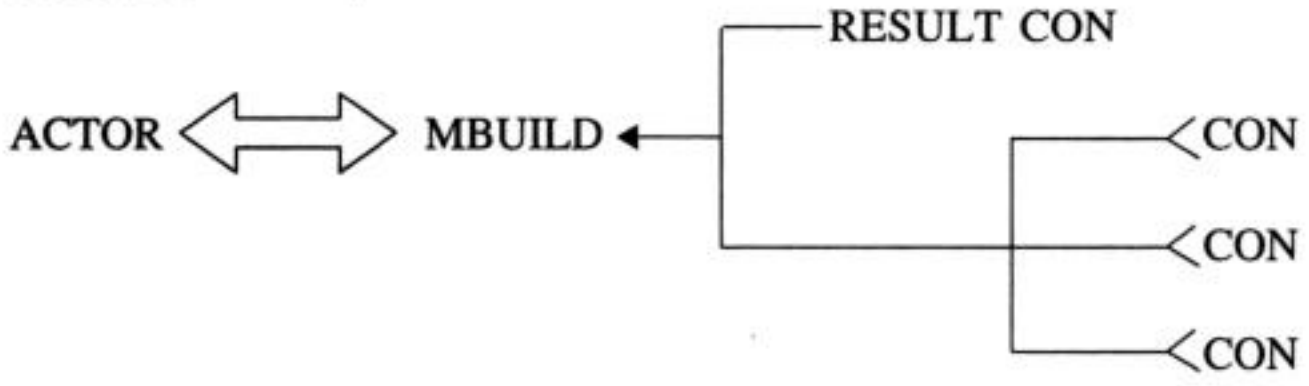
An example of MTRANS is given below.

You remember Jake was a driver.



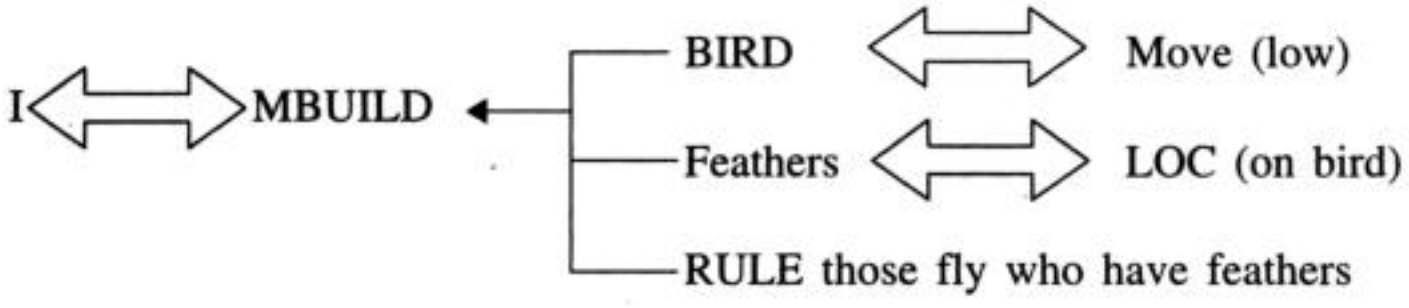
MBUILD: MBUILD is an ACT underlying the verb and it accounts for thought combination on, and there is no conceptualization such as think ORS, consider, resonant, resolve, conclude, solve, relate, etc.

It is written as

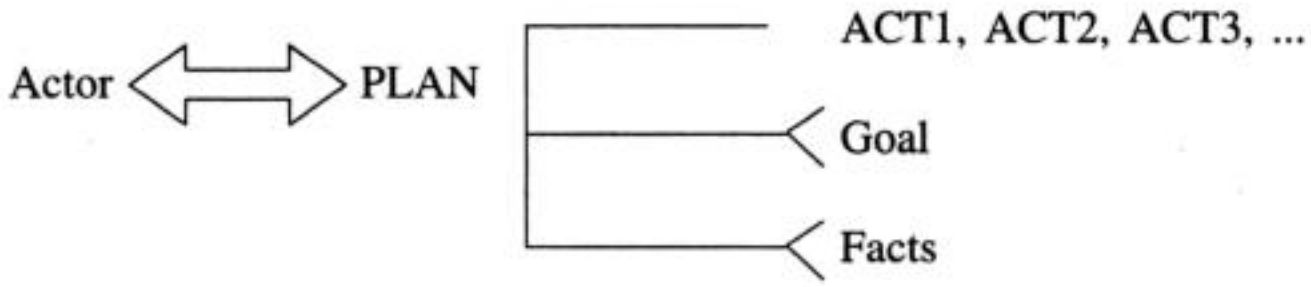


It takes as object a many-to-one 'functional' arrow that denotes the combination and transformation of several units into one resultant unit. For example, the sentence:

"I guessed that bird would fly because it has feathers" can be represented as such



PLAN: One's ability for step-by-step course of action, which leads to the goal, is measured by PLAN. It takes input as a goal and a set of possible plans that actor includes to act out in order to achieve the desired goal. This structure of PLAN is shown below:



Conceptual Tenses

The tense, mood or other aspects of a verb in the sentence can be described by conceptual tenses proposed by Schank and Riesbeck (1973) which are as follows:

P = Past, F = Future, t = transition
Ts = Start transition, tf = Finished transition, k = continuing
? = Interrogative, /= Negative, Nil = Present
delta = Timeless, C = Conditional

SCRIPTS

A script is a structure that is used to describe stereotyped events and also the prototypes of expected sequences of events. These events are likely to occur in a given situation or context. These are basically a frame like structure and script of a play, it is described in terms of actors, roles, props and scenes. It consists of a set of slots like a scene of a play. Some kind of information is associated with each slot. The various components of the script are as follows:

Entry conditions: Some initial conditions are to be set before the occurrence of events described in a script.

Result: After the completion of the events, the conditions should be true.

Props: In the description of the events in the script, these are the slots representing the objects in the event.

Roles: Slots represent the people involved in the actions of the script.

Track: The specific variation on a more general pattern is represented by this particular script.

Scenes: The sequence of events that occurs apart from the above components one can include entries regarding time factors, place of occurrence or points of view.

A slot in a script, which corresponds to the parts of the events, is filled with primitives of conceptual dependency structure. These are shown in Table 6.1, which is a script model of the events corresponding to the description of a visit to party in a hotel.

Visit to a party in a hotel

Prop: Garage Car Hotel	Scene 1:	Start up Person reaches garage Person takes out his car Person closes his garage (if not automatic) Person comes out of the compound Person moves on the road
Roles: Person Host	Scene 2:	On drive Person sets drive on the highway Person tracks down to road Person reaches the hotel
Entry Condition Some one to host a party	Scene 3:	In the hotel Person gets car parked Person reaches the reception counter Person is welcome by the host.
	Scene 4:	In the party The person is escorted to the drink corner Person is invited to dance Person is involved with the chitchat Person is requested to dine Person bids goodbye to the host Person returns in jubilant mood.
	Scene 5:	Back to house Person gets back to the car Person sets on drive Person drives on the highway Person enters the town Person enters his house Person parks car in his garage. Person finds himself in his house.

Reasoning with Script

The reasoning process involves finding the S best-fit match between the observed conditions and the entry conditions of the script.

On the basis of certain observed events in a particular scene, one can infer unobserved events. For example, a person is welcomed by the host in Scene 2 or 3, one may infer that the party is certainly going to take place at the scheduled venue. But it may not be true, the host shows overwhelmed welcome and will beg excuses in due change of venue.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

EXERCISES

6.1 Distinguish between the methods of FOL and Prolog for Knowledge Representation on the basis of syntax, semantics and variables.

6.2 Represent the following sentences in PROLOG:
: Som is a computer science student.
: He does not sing.
: Jane jumps well.
: She is a player.
: She plays well and dances well.
: Som and Jane are close friends.

6.3 Develop a rule-base model in PROLOG indicating the proper domain, database, predicates, clauses and rules for following facts:

A computing system with a networking to be developed for different purposes such as laboratory work, information processing, research work, scientific computation and course instruction. One has to select different configuration of a computing system and a particular topology and protocol of networking.
Make a selection procedure for the above purpose in PROLOG.

6.4 A: Discuss backtracking mechanism in the Problem no. 6.3 step by step.
B: What are static and dynamic databases in PROLOG. How are these implemented?

6.5 For the three types of query, numeric, binary and text; develop a program in PROLOG.

QUERY1: Q: What is your age?
A: 42.
QUERY2: Q: Are you young?
A: Yes.
QUERY3: Q: What is your name?
A: My name is Fyoder.

6.5 Develop a script model of the events: Starting with bike from the hostel, then to the main road, and then to the classroom and finally return to hostel.

Chapter 7



Automated Reasoning

Automated reasoning is concerned with the real world problems that have imperfect information exhibiting the attributes as uncertain, imprecise, inconsistent, and ambiguous or some combination thereof. There are various approaches to handle such situations based upon logic and measures or some combination of these two. The measures may be confidence factor, probability, belief or degree of membership in a set. The logic may be predicate calculus, abduction and inheritance.

Default reasoning combines the properties and approaches of many reasoning methods. Reiter (1980) proposed the default reasoning that incorporated heuristic assumptions and default into the reasoning process. Non-monotonic logic is also one of the approaches to deal with imperfect information. In this chapter, default logic, closed word assumption, circumscription, model-based and case-based reasoning methods have been discussed. In this context, some of the problems which have the characteristics of imperfect information are also discussed.

DEFAULT LOGIC

Default logic was introduced by Reiter (1980) and is one of the most extensively used non-monotonic formalisms. The default theory is defined as a pair (D, W), where D is a set of default and W is a set of formula. D is also a collection of special inference rule called defaults. A default is a rule of the form: Δ

$$(\alpha(x): M\beta(x))/\gamma(x) \text{ or } (\alpha(x): \beta_1(x) \ldots \beta_n(x))/\gamma(x)$$

where $\alpha(x)$, $\beta(x)$, and $\gamma(x)$) are well-formed formula in first order logic whose free variables are $x = (x_1 \ldots x_n)$. $\alpha(x)$ is called the prerequisite, $\beta(x)$ is the justification and $\gamma(x)$ is the consequent of δ^*. JUST (D) denotes the set of all predicate symbols appearing for a set D of defaults in the justification of defaults in D. The default D is called closed if none of $\alpha(x)$, $\beta(x)$ and $\gamma(x)$contains free variables usually represented by α: $M\beta/\gamma$. Otherwise, it is called open.

When $n = 0$, then the proportional constant T (true) is implicitly assumed as the justification of the defaults. A default theory is closed if all the defaults in D are closed. The

Copyrighted material

defaults $(\alpha(x)\colon M\beta(x))/\gamma(x)$ state that for all variables in x if the prerequisite $\alpha(x)$ is believable, and the justification, $\beta(x)$ is not believable, then the consequent $\gamma(x)$ can be assumed.

The semantic of closed default theory (D, W) is based on the notion of the extension. Δ is closed when every default in D is used and every formula in W is closed. Otherwise, it is open. It will be quantifier free if every default in D is quantifier free and every formula in W is quantifier free. A closed quantifier free default theory is caused by ground default theory.

Extension

Definition of extension of default theories and generation defaults was originally introduced by Reiter (1980). Each extension E of (D, W) is identified by its generating defaults which allow for a compact representation of E. The generative default of E is defined by $GD(E, (D, W))$ are the defaults $(\alpha\colon \beta_{1\ldots}\ \beta_n)/\gamma$ form D such that $\alpha \in E$ and $\neg\beta_j \notin E$ for all $j = 1, \ldots, n$, then E is constructible from W and $GD\ (E, (D, W))$ as follows.

If E be an extension of the closed default theory

$$\Delta = <D, W> \text{ then}$$

$$E = \text{cons}\left[W \cup \left\{\frac{\alpha : \beta_1, \ldots, \beta_n}{\gamma} \in GD(E, \Delta)\right\}\right]$$

The definition of extension is extended to D default theories by assuming that the default with free variables implicitly stand for the infinite set of closed defaults obtained by replacing the free variables with the terms of the Herband universe of the default theory.

If $\Delta = (D, W)$ is an open quantifier, is free default theory and E is a set of quantifier, is free ground formula.

Then define

$E^{(0)} = \text{INST}\ (W)$;

$E^{(i+1)} = \text{Th}(E^{(i)}) \cup \{\gamma/\alpha\colon M\beta/\gamma \in \text{INST}(D)$,

where $\alpha \in E^{(i)}$ and $\neg\beta \notin E\}$

E is an extension for Δ iff $E = \cup_{i \geq 0} E^{(i)}$

INST(W) and INST(D) denote the result of instantiation of every free variable in W and Δ with a ground term, respectively.

$(E^{(i)})$ denotes the set of all logical consequent from $E^{(i)}$. A formula is said to be believable in a default theory if it is contained in some exclusion of the default theory. For $\Delta = (D, W)$ an open quantifier-free default theory and E is an extension of Δ, the set of generation defaults w.r.t. E denoted by $GD(E)$ is given by

$GD(E) = \{\delta/\delta = \alpha\colon M\beta/\gamma \in \text{INST}(D)$, where $\alpha \in E$ and $\neg\beta \notin E\}$

$\Delta' = ((GD(E)), \text{INST}(D))$ is the generalization theory w.r.t. E.

By definition, every default in $GD(E)$ is a ground default, i.e. a closed quantifier default.

Example 7.1 A well-known example of default theory is as follows:

Assume that

$$D = \{Q: P/P, R: \neg P/ \neg P\}, W = \{P, Q\}$$

where P, Q and R are propositional atoms, with meaning that someone is pacifist, a quaker, and republican, respectively. Two extensions of the default theory $\Delta = (D, W)$ are: Cons ($\{P, Q, R\}$) and cons ($\{\neg P, Q, R\}$). The first exclusion states that a republican is a quaker as well as pacifist, whereas the second exclusion states that the republican is a quaker but not a pacifist. The first exclusion has the (single) generating default Q: P/P the second R: $\neg P/\neg P$. The credulous semantics conduces that the person is a pacifist as well as not a pacifist. The cautious semantics support none of the conclusion.

Application in query language

Marcoli et al. (1997) have used default as query language, naming *DQL* for finite relational databases. They have illustrated several examples for electric circuit, company's control strategies and maximal trust, etc. An example is given to make the understanding of the default logic more clear.

Suppose, two relation scheme MARRIED and STUDENT have a single attribute NAME. The *I/O* query is specified by (B, D) is used to compute instances of the output relations ADULT (NAME) and EMPLOYEE (NAME).

$B = \{\text{married}(x)\}\ \text{adult}(x)\}$,

$$D = \left\{ \frac{\text{student}(x): \text{employee}(x)}{\neg\text{employee}(x)}, \frac{\text{adult}(x): \text{employee}(x)}{\text{employee}(x)} \right\}$$

The basic assumption is that students are generally not married and are also not employed but adults are usually employed.

Consider the following instance W of R when the domain is (Sam, Jane, Patrick).

MARRIED

NAME
JANE

STUDENT

NAME
SAM
PATRICK

The query computes the following output

ADULT

NAME
JANE

EMPLOYEE

NAME
JANE



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

III. $W\ \phi \rightarrow \varphi$ denotes a formula φ is a consequence of ϕ in W iff each preferred state s for ϕ is such that $f(s)\ \varphi$.

Z framework

Z framework of reasoning was developed by Pearl (R) based on a ranking of default rules that deal with the notion of specificity.

Between a default rule $\alpha \rightarrow \beta$ and the possible world W, the following three relations are formed:

- **Verification:** This world verifies the rule if $W\ \alpha \wedge \beta$.
- **Satisfaction:** This world satisfies the rule if $W \vdash \alpha \vee \beta$.
- **Falsification:** This world falsifies the rule if $W\ \alpha \wedge \vdash \beta$.

  The rank order of rules in given default base $\Delta = \{\alpha_i \rightarrow \beta_i \mid i = 1, \ldots, m\}$, is made in such a way that the rules with general premise get (least specific) the least priority. A notion of tolerance was introduced by Pearl such as a rule $\alpha \rightarrow \beta$ is said to be tolerated by a base $\Delta = \{\alpha_i \rightarrow \beta_i \mid i = 1, \ldots m\}$, iff $\alpha \wedge \beta, \vdash \alpha_i \vee \beta_i, \ldots, \vdash \alpha_m \vee \beta_m$ are consistent.
- Δ is partitioned in a ordered set (stratification) $\{\Delta_1, \Delta_2, \ldots, \Delta_k\}$ such that rules Δ_i are tolerated by all rules in $\Delta_i \cup \ldots \cup \Delta_k$.
- A ranking Z correspondingly order the rules of Δ is defined such that $Z(d) = I$ for each rule d in Δ_i.

  The non-monotonic consequence relation Z is defined by building a ranking function K on the words from Z ranking by $K(w)= \max\{Z(d_i) + 1 \mid w\ \alpha_i \wedge \beta_i,\ d_i = \alpha_i \rightarrow \beta_i, \in \Delta\}$ and $K(w) = 1$ if w satisfies all the rules of Δ.
- A ranking function formula Z is induced by K

$$Z(\alpha) = \min\{K(w) \mid w\ \alpha\}$$

If the models of α have lower rank than $Z(\alpha)$, then it denotes the degree of 'abnormality' of α, with respect to the rule in Δ.Z is disbelief function. Now the Z relation is defined by

$$]\alpha\ \beta \text{ iff } Z(\alpha \wedge \beta) < Z(\alpha \wedge \vdash \beta),[$$

Penalty logic

- Pinkas proposed penalty logic and was used by Benferhat and Dupin (1996) for modelling the inconsistency in the knowledge base. It states:

  Each formula of a $KB\ \Sigma$ is associated with a number, called penalty, representing a kind of price. If a formula is not satisfied, a price is charged in such a way that the sum of penalties (charged price) for the unsatisfied an interpretation cost induce a preference over interpretation.

  For each price of information, i.e. default Δ with satisfaction $(\Delta_1, \Delta_2, \ldots, \Delta_n)$ a penalty $C(d) = I$ is associated with each default d in Δ_i, if d is unsatisfied. Based on the cost, the compute order on the elements of Ω is induced by these penalties such as:

$$C(w)= \Sigma\{C(d) \mid d \in \Delta\ s \cdot t\ w\cdot\} =$$



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

Order-based retrieval: The order-based retrieval offers an expressive query language for defining and combining ordering relations, and the result of query evaluation is to partially order the cases in the case base. The query language supports queries that naturally combine preferred values with other preference information. There is no need for an explicit measure of recommendation diversity.

Explanation-oriented retrieval: The explanation-oriented retrieval is a precedent-based approach, but once a classification or diagnosis has been reached on the basis of the nearest neighbours, the system performs an additional retrieval step, using an explanation utility metric to obtain the explanation case. These explanation cases are more compelling than the nearest neighbour (Doyle, 2004).

Adaptation

Single-case adaptation

If only the most similar case is taken as the adaptation object, the adaptation type is named as single-case adaptation. It can be classified as follows:

Transformational adaptation: Transformational adaptation reorganizes the elements. It can revise, add, delete and change solution elements under certain conditions. According to the structure of solution, it can be classified into two categories substitutional adaptation and structural adaptation.

Substitutional adaptation: In substitutional adaptation, only the parameter of solution is changed, and the structure of solution is not changed. Transformational adaptation needs the knowledge on how the attributes difference leads to different solution.

Structural adaptation: In structural adaptation, rules are directly applied to the solution stored in cases (Kolodner, 1989). In this method some operators such as adding, deleting, changing and so on are adopted. This technique is used in JUDGE (Bain and Muggleton, 1991) and CHEF (Hammond, 1987).

Compositional adaptation: In compositional adaptation, the solution element (set) is obtained by combining solution elements of several partial similar cases, as in CAPLAN/CBC. It is not the simple combination of above adaptation types. Due to the complexity of problems, sometimes similar cases cannot be retrieved. Now, the case attribute set is divided into several independent subsets, then for similar cases according to subset can be retrieved, and the solution elements according to different subset can be obtained. Finally, the solution elements are combined a new solution.

Derivational adaptation: This method reuses the algorithms, methods or rules that are used to generate the original solution for producing a new solution to the current problem. Derivational adaptation needs the knowledge of generating solution from scratch. It stores the



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

- **Case-based substitution:** This method uses cases to suggest solution adaptation. This technique of adaptation is used in ACBARR (Moorman and Ram, 1992), a system for robot navigation.

MULTIMODAL REASONING

Multimodal reasoning (MMR) is an approach that integrates different types of knowledge in the same decision-support system. The logic behind the strategy of integrating different kinds of knowledge is that the knowledge for a given domain and given task has to be modelled in different ways because it is inherent at different levels.

The two types of knowledge are: explicit knowledge, and implicit knowledge. Explicit knowledge is the knowledge which is already well established and formalized domain knowledge. This knowledge can be present in books and in written documentation, or can be represented by some formalisms for developing knowledge-based systems, for example, taxonomies or rules, or finally by mathematical or probabilistic models. While implicit or tacit knowledge consists of individual expertise, organizational practices, and past cases. It can be represented by heuristics that even experts are not fully aware of, or can be stored in databases and case libraries. Implicit knowledge is managed by a methodology case-based reasoning (CBR) and explicit knowledge is managed by two most suitable methodologies rule-based reasoning (RBR) and model-based reasoning (MBR).

The different types of combination in multimodal reasoning are:

- Combination of rule-based reasoning with case-based reasoning.
- Combination of case-based reasoning and multimodel reasoning.

Combination of Rule-Based Reasoning and Case-Based Reasoning

In the development of expert systems, symbolic rules play an important role in knowledge representation. They represent general knowledge of the domain and exhibit a number of attractive features such as naturalness, modularity and ease of explanation. The major drawback of them is the difficulty of acquiring them. The traditional process of eliciting rules through the interaction with the expert may turn out to be a bottleneck because it causes delays in the system's overall development and the acquired rules may be imperfect and not covering the full complexities of the domain. To deal with these advantages rule induction methods are highlighted but they are still unable to recognize exceptions in small, low frequency sections of the domain. Finally, rules may contain terms that are not well-defined. These drawbacks are partially overcome by case-based reasoning (CBR). The specific feature of CBR are:

- Cases represent specific knowledge of the domain.
- They are natural and usually easy to obtain.
- New cases can be easily inserted into a knowledge base without making changes to the pre-existing knowledge.
- Incremental learning comes natural to case-based reasoning.
- The more the cases are available the better will be the domain knowledge represented.

The accuracy of a case-based system can be enhanced throughout its operation, as new cases become available. Cases also have a negative aspect as compared to symbolic rules, that is, they do not provide concise representations of the incorporated knowledge. Furthermore, the time-performance of the retrieval operations is not always desirable.

To deal with the drawbacks of CBR and rule-based reasoning (RBR), an integrated approach of CBR and RBR is used. Integrating rule-based and case-based reasoning have given interesting and effective knowledge representation schemes. The goal of these efforts is to derive hybrid representations that augment the positive aspects of the integrated formalisms and simultaneously minimize their negative aspects. RBR's main strength is that it does not require a lot of space for storing all the generalized rules. CBR in the other hand, allows the reasoner to propose solutions to problems really fast-based on the stored cases. CBR also allows the reasoner to propose solutions in domains that are not completely understood. Furthermore, it allows the avoidance of previous mistakes.

ABDUCTION

C.S. Peirce, a logician and mathematician, defined abduction in the following way:

> "The first stating of a hypothesis and the entertaining of it, whether as a simple interrogation or with any degree of confidence is an inferential step which I propose to call abduction or retroduction" and "nothing justifies a retroduction inference except it affording an explanation of the facts."

The adductive method has been applied (Punch (III) et al., (1990)), on the red systems for blood antibody identification. They had developed a tool called PIERCE which enables knowledge engineers to determine when a method is appropriate for invocation and how the process selects a method for invocation. The salient features are as follows:

- The adductive assembly algorithm contains three main modules, viz. plausibility, evaluation, assembly and entique.
- The hierarchical classification is used to produce eventually a list of antibodies and their associated plausibility. It passes this list to assembly module as a set of candidate hypothesis to be used in explaining test panels' reactions.
- The three requirements—coverage, plausibility and compatibility—lead to following algorithms:
  - Single finding required for explanation is selected,
  - Select those hypotheses that offer explanation from the antibody hypotheses found by hierarchical classifier,
  - Select only the one hypotheses which offers to explain the finding otherwise most plausible,
  - Integrate the selected hypothesis into the compound explanation,
  - Update any other findings that the integration might explain and if some findings still remain unexplained, go to the initial step.

Abduction is non-deductive form of inference. Given the truth of proposition of Q and the implication $P \rightarrow Q$, one gets the conclusion P. If two propositions: there is a smoke Q and there would be fire P is written in casual relationship as $P \rightarrow Q$ by abduction, then there is smoke it is concluded that there would be fire. But presence of smoke may not be caused by fire, it may be smoke gas. Thus, abduction does not guarantee the definite conclusion.

The abduction inference is represented in the following way:

$$(P \overset{c}{\rightarrow} Q) / P$$

Where Q is assertion $P \overset{c}{\rightarrow} Q$ is an implication where c over the implication arrow means a possible casual relationship between P and Q. P is the conclusion obtained through abduction.

TRUTH MAINTENANCE SYSTEM (TMS)

The truth maintenance system is concerned with maintaining truth in the system which revises the set of benefits when new information contradicts existing information. The TMS works with the inference engines (IE) that acts as a problem solver. In a large search space, the IE uses different search strategies for alternative paths or choices and examines the consequence of each path. If a contradiction is found during this process, the TMS eliminates it by reusing the knowledge base. The IE and TMS, together can solve the problem where algorithm solution does not exist. However, the IE cannot derive valid conclusion if it uses data from different problem states or contexts. A typical structure of TMS-based knowledge-based system is shown in Figure 7.3.

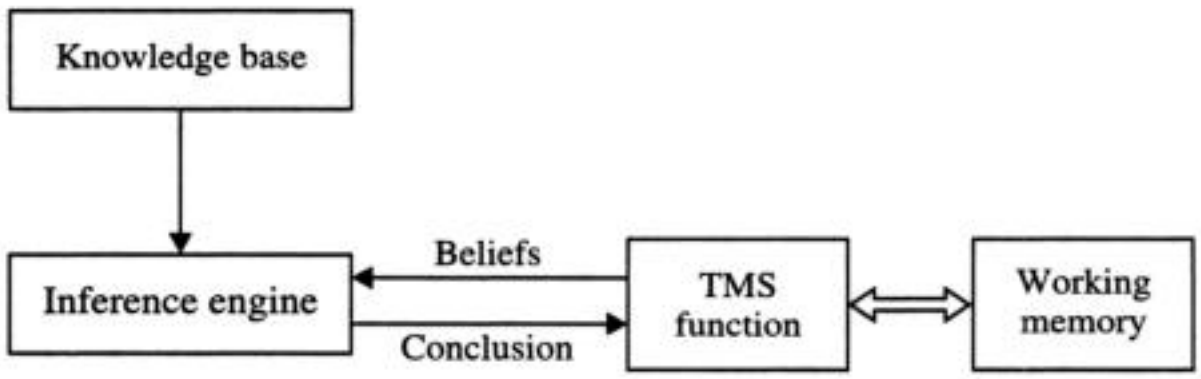


Figure 7.3 Structure of TMS-based KBS.

The knowledge base contains the description of problem in terms of facts, figures and their relationships which is also the initial problem state in the working memory. The working memory contains the dynamic data that reflect problem data transmission. The inference engine calls TMS function to obtain data required in reasoning coupled with belief reason. The number of TMS functions that must be implemented depending on the complexity of each such problem. The minimal TMS configuration has function that performs belief revision to determine which data hold when a context switch occurs. TMS functions that perform non-monotonically handling must be used to enable solution of problem with in couple information.

In the justification-based TMS, each node has association with justification for the node's truth value. In TMS, a node represents one unit of knowledge, a fact, rule assertion, etc. At any

point in execution, every node is one of two conditions IN: currently believed to be false. Justification for the truth node's truth value is associated with each node. Support list (SL) is the simplest form of justification which is of the form [SL (in-nodes){out-nodes)]. Where in-nodes is the list of all nodes that must be; IN for this node to be true and (out-node) is the list of all nodes that must be out for this node to be true. For example, the following statements or assertions are given, (i) It is a house, (ii) It is a hotel, (iii) It is personal, (iv) It is public, (v) It is an enterprise. Then support list for the above statements are developed as shown in Table 7.3.

Table 7.3

Node number	*Proposition*	*Justification*
1	It is a house	[SL (0)(2)]
2	It is a hotel	[SL (0)(2)]
3	It is a personal	[SL (1)(2)]
4	It is public	[SL (2)(1)]
5	It is enterprise	[SL (2)(1)]

Assertions (i) and (ii) are premises which do not have IN, and (iii), (iv), (v) are derived facts. Premises in principle are the basic assumption to start with and, hence, they do not have IN list.

The working principle of JTMS can be illustrated with an example. Suppose Jane wishes to go for picnic as it is fine weather, but when she sets out for the purpose weather becomes cloudy and it begins to rain. Then she changes her plan and goes to gymnasium. In the evening she becomes tired and she went to hostel to relax. In the given description of situation we derive the proposition and supports the list for the justification on the following manner:

Node number	Proposition	Justification
1	Normal weather	[SL ( ) (2)]
2		
3	Raining	
4	Not raining	
5	Play time	[SL ( ) (5)]
6	Evening	[SL ( ) (6)]

A problem solver concludes using assumptions (i) and (ii) that the weather is fine. It provides a justification for node 7 which makes this node (n), i.e.

7: fine weather. [SL(1,4) ( )].

It is time for picnic is inferred by node 7 and 5 and is directed for providing corresponding justification for node 8 which takes IN in SL.

8: Picnic [SL (5,7), ( )];

The contradiction arises in the situation when it begins to rain which leads to the introduction of contradiction node 9.

9: Contradiction [SL 4,( )].



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



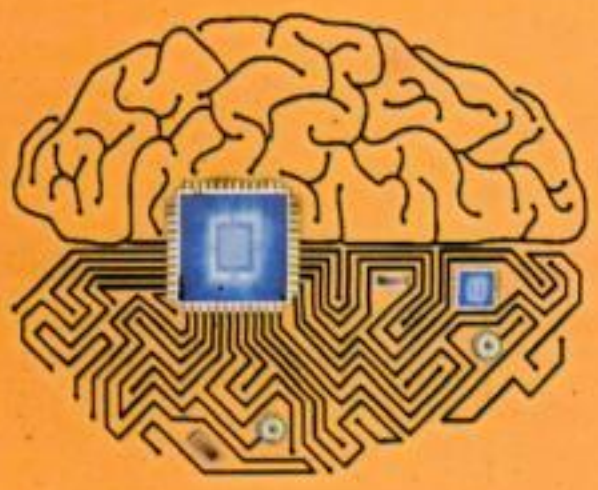
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



Artificial Intelligence

R.B. Mishra

This book has been written keeping in view the requirements of undergraduate and postgraduate students and research scholars in the area of computer science and engineering in particular, and other branches of engineering which deal with the study of AI such as electronics engineering, electrical engineering, industrial engineering (robotics and FMS). Besides the engineering students, the postgraduate students of computer science and computer applications and cognitive sciences researchers can equally benefit from this text.

The basic concepts of artificial intelligence, together with knowledge representation, reasoning methods, acquisition, management and distributed architecture, have been nicely and instructively described. The various application domains and disciplines in engineering, management, medicine which cover different aspects of design, assembly and monitoring, have been presented with utility aspects of AI concepts in logic and knowledge.

The book maintains a simple and comprehensible style of presentation for the different categories of readers such as students, researchers and professionals for their respective uses.

You may also be interested in

Neural Networks, Fuzzy Logic, and Genetic Algorithms: Synthesis and Applications (with CD-ROM),
S. Rajasekaran, G.A. Vijayalakshmi Pai

Artificial Neural Networks, B. Yegnanarayana

Fundamentals of Artificial Neural Networks,
Mohamad H. Hassoun

Foundations of Robotics: Analysis and Control,
Tsuneo Yoshikawa

Introduction to AI Robotics, Robin R. Murphy

Mechanics of Robotic Manipulation, Matthew T. Mason

₹ 325.00

www.phindia.com

THE AUTHOR

R.B. MISHRA, Ph.D., is Professor, Department of Computer Engineering, Institute of Technology, Banaras Hindu University, Varanasi. He has 30 years of teaching and research experience. He has presented and published more than 100 research papers and articles in conferences and journals of national and international repute. His areas of interest include artificial intelligence and multi-agent systems and their applications to semantic web, e-commerce, medical computing, machine translation, intelligent tutoring system, and robotics.

ISBN:978-81-203-3849-4

9 788120 338494

Copyrighted material