NOTES ON ARTIFICIAL INTELLIGENCE

PREPARED BY:

Mr. B P Mishra

Rajdhani College of Engineering And Management,
BHUBANESWAR

ARTIFICIALINTELLIGENCESYLLABUS

Module1 **12Hrs**

What is Artificial Intelligence? AI Technique, Level of the Model,Problem Spaces, and Search: Defining the Problem as a State Space Search, Production Systems, Problem Characteristics, Production System Characteristics, Issues in the Design of SearchPrograms. Heuristic Search Techniques: Generate-and- Test, Hill Climbing, Best-first Search, Problem Reduction, Constraint Satisfaction, Means-ends

Analysis, Knowledge Representation: Representations and Mappings, Approaches to Knowledge Representation, Using Predicate Logic: Representing Simple Facts in Logic, Representing Instance andISA Relationships, Computable Functions and Predicates, Resolution, Natural Deduction.Using Rules: Procedural Versus Declarative Knowledge, Logic Programming, Forward Versus Backward Reasoning,

Matching, Control Knowledge.Symbolic Reasoning Under Uncertainty: Introduction to Nonmonotonic Reasoning, Logics for Nonmonotonic Reasoning, Implementation Issues, Augmenting a Problem-solver, Depth-first Search, Breadthfirst Search.Weak and Strong Slot-and-Filler Structures: Semantic Nets, Frames, Conceptual DependencyScripts, CYC.

Module2 **10Hrs**

GamePlaying: TheMinimax SearchProcedure,AddingAlpha-betaCutoffs,IterativeDeepening.Planning: The Blocks World, Components of a Planning System, Goal Stack Planning, Nonlinear Planning Using Constraint Posting, Hierarchical PlanningOther Planning Techniques.Understanding: What is Understanding, What Makes Understanding Hard?, Understanding as Constraint Satisfaction.Natural Language Processing: Introduction, Syntactic Processing, Semantic Analysis, Discourse and Pragmatic Processing, Statistical Natural Language Processing, Spell Checking.

Module3 **8Hrs**

Learning: Rote Learning, learning by Taking Advice, Learning in Problem-solving, Learning fromExamples: Induction, Explanation-based Learning, Discovery, Analogy, Formal Learning Theory, Neural Net Learning and Genetic Learning. Expert Systems: Representing and Using Domain Knowledge, Expert System Shells, Explanation, Knowledge Acquisition.

TextBook:

1. ElaineRich,KevinKnight,&ShivashankarBNair,ArtificialIntelligence, McGrawHill,3rded.,2009 References:
1) IntroductiontoArtificialIntelligence&ExpertSystems,DanWPatterson,PHI.,2010

2) SKaushik,ArtificialIntelligence,CengageLearning,1sted.2011

Module1

ARTIFICIALINTELLIGENCE

WhatisArtificial Intelligence?

It isa branchof ComputerScience that pursues creatingthe computers or machines asintelligent as human beings.

It is the science and engineering of making intelligent machines, especially intelligent computer programs.

It is related to the similar task of using computers to understand human intelligence, but **AI**does not have to confine itself to methods that are biologically observable

Definition: Artificial Intelligence is thestudyof howto make computers do things, which, at the moment, people do better.

According to the father of Artificial Intelligence, John McCarthy, it is *"The science and engineering of making intelligent machines, especially intelligent computer programs".*

Artificial Intelligence is a way of **making a computer, a computer-controlled robot, or a software think intelligently**, in the similar manner the intelligent humans think.

AI is accomplished by studying how human brain thinks and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.

It has gained prominence recently due, in part, to big data, or the increase in speed, size and variety of data businesses are now collecting. AI can perform tasks such as identifying patternsinthedatamoreefficientlythanhumans,enablingbusinessestogainmoreinsightoutof their data.

From a **business** perspective AI is a set of very powerful tools, and methodologies for using those tools to solve business problems.

From a **programming** perspective, AI includes the study of symbolic programming, problem solving, and search.

AI Vocabulary

Intelligence relates to tasks involving higher mental processes, e.g. creativity, solving problems, pattern recognition, classification, learning, induction, deduction, building analogies, optimization, language processing, knowledge and many more. Intelligence is the computational part of the ability to achieve goals.

Intelligent behaviour is depicted by perceiving one's environment, acting in complex environments, learning and understanding from experience, reasoning to solve problems and discover hidden knowledge, applying knowledge successfully in new situations, thinking abstractly, using analogies, communicating with others and more.

Science based goals of AI pertain to developing concepts, mechanisms and understanding biological intelligent behaviour. The emphasis is on understanding intelligent behaviour.

Engineering based goals of AI relate to developing concepts, theory and practice of building intelligent machines. The emphasis is on system building.

AI Techniques depict how we represent, manipulate and reason with knowledge in order to solve problems. Knowledge is a collection of 'facts'. To manipulate these facts by a program, a suitable representation is required. A good representation facilitates problem solving.

Learning means that programs learn from what facts or behaviour can represent. Learning denotes changes in the systems that are adaptive in other words, it enables the system to do the same task(s) more efficiently next time.

Applications of AI refers to problem solving, search and control strategies, speech recognition, natural language understanding, computer vision, expert systems, etc.

ProblemsofAI:

Intelligence does not implyperfect understanding; everyintelligent beinghas limited perception, memory and computation. Many points on the spectrum of intelligence versus cost are viable, from insects to humans. AI seeks to understand the computations required from intelligent behaviour and to produce computer systems that exhibit intelligence. Aspects of intelligence studied byAIinclude perception, communicational usinghuman languages, reasoning, planning, learning and memory.

Thefollowingquestionsaretobeconsideredbeforewecan step forward:
1. Whataretheunderlyingassumptionsaboutintelligence?
2. Whatkinds of techniques will beuseful for solvingAIproblems?
3. Atwhatlevelhumanintelligencecanbemodelled?
4. Whenwillitberealizedwhenanintelligentprogramhasbeen built?

BranchesofAI:

A list of branches of AI is given below. However some branches are surely missing, because no one has identified them yet. Some of these maybe regarded as concepts or topics rather than full branches.

Logical AI — In general the facts of the specific situation in which it must act, and its goals are all represented by sentences of some mathematical logical language. The program decides what to do by inferring that certain actions are appropriate for achieving its goals.

Search — Artificial Intelligence programs often examine large numbers of possibilities – for example, moves in a chess game and inferences by a theorem proving program. Discoveries are frequently made about how to do this more efficiently in various domains.

Pattern Recognition — When a program makes observations of some kind, it is often plannedto compare what it sees with a pattern. For example, a vision program maytryto match a pattern of eyes and a nose in a scene in order to find a face. More complex patterns are like a natural language text, a chess position or in the history of some event. These more complex patterns require quite different methods than do the simple patterns that have been studied the most.

Representation —Usuallylanguagesofmathematicallogicareusedtorepresentthefactsabout the world.

Inference — Others can be inferred from some facts. Mathematical logical deduction is sufficient for some purposes, but new methods of *non-monotonic* inference have been added to the logic since the 1970s. The simplest kind of non-monotonic reasoning is default reasoning in which a conclusion is to be inferred by default. But the conclusion can be withdrawn if there is evidence to the divergent. For example, when we hear of a bird, we infer that it can fly, but this conclusion can be reversed when we hear that it is a penguin. It is the possibility that aconclusion may have to be withdrawn that constitutes the non-monotonic character of the reasoning. Normal logical reasoning is monotonic, in that the set of conclusions can be drawn from a set of premises, i.e. monotonic increasing function of the premises. Circumscription is another form of non-monotonic reasoning.

Common sense knowledge and Reasoning — This is the area in which AI is farthest from the human level, in spite of the fact that it has been an active research area since the 1950s. While there has been considerable progress in developing systems of *non-monotonic reasoning* and theories of action, yet more new ideas are needed.

Learning from experience — There are some rules expressed in logic for learning. Programs can onlylearn what facts or behaviourtheirformalisms can represent, and unfortunatelylearning systems are almost all based on very limited abilities to represent information.

Planning — Planning starts with general facts about the world (especially facts about the effects of actions), facts about the particular situation and a statement of a goal. From these, planning programs generate a strategy for achieving the goal. In the most common cases, the strategy is just a sequence of actions.

Epistemology — This is a study of the kinds of knowledge that are required for solvingproblems in the world.

Ontology — Ontology is the study of the kinds of things that exist. In AI the programs and sentences deal with various kinds of objects and we study what these kinds are and what their basic properties are. Ontology assumed importance from the 1990s.

Heuristics — A heuristic is a way of trying to discover something or an idea embedded in a program. The term is used variously in AI. *Heuristic functions* are used in some approaches to search or to measure how far a node in a search tree seems to be from a goal. *Heuristicpredicates* that compare two nodes in a search tree to see if one is better than the other, i.e. constitutes an advance toward the goal, and may be more useful.

Genetic programming — Genetic programming is an automated method for creating a working computer program from a high-level problem statement of a problem. Genetic programming starts from a high-level statement of 'what needs to be done' and automatically creates a computer program to solve the problem.

ApplicationsofAI

AIhasapplicationsinallfieldsofhumanstudy,suchasfinanceandeconomics,environmental engineering, chemistry, computer science, and so on. Some of the applications of AIare listed below:

- Perception
    - ■ Machine vision
    - ■ Speechunderstanding
    - ■ Touch (*tactile*or*haptic*) sensation
- Robotics
- NaturalLanguageProcessing
    - ■ NaturalLanguageUnderstanding
    - ■ SpeechUnderstanding
    - ■ LanguageGeneration
    - ■ MachineTranslation
- Planning
- Expert Systems
- MachineLearning
- TheoremProving
- SymbolicMathematics
- Game Playing

AI Technique:

Artificial Intelligence research during the last three decades has concluded that *Intelligence requiresknowledge*.Tocompensateoverwhelmingquality,knowledgepossesseslessdesirable properties.

A. Itishuge.
B. Itisdifficulttocharacterize correctly.
C. Itisconstantlyvarying.
D. Itdiffers from databybeingorganized inawaythatcorresponds toitsapplication.
E. Itiscomplicated.

AnAItechniqueisa method thatexploits knowledgethatis represented sothat:

- Theknowledgecapturesgeneralizationsthatshareproperties,aregrouped together, rather than being allowed separate representation.

- Itcanbeunderstoodbypeoplewhomustprovideit—eventhoughformany programs bulk of the data comes automatically from readings.

- InmanyAIdomains,howthepeopleunderstandthesamepeoplemustsupplythe knowledge to a program.

- Itcanbe easilymodifiedtocorrecterrorsandreflectchangesinrealconditions.

- Itcan bewidelyused even if it is incomplete or inaccurate.

- Itcanbeusedtohelpovercomeitsownsheerbulkbyhelpingtonarrowthe range of possibilities that must be usually considered.

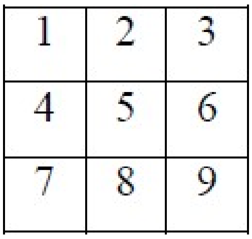
In order to characterize an AI technique let us consider initially OXO or tic-tac-toe and use a series of different approaches to play the game.

The programs increase in complexity, their use of generalizations, the clarity of their knowledge and the extensibility of their approach. In this way they move towards being representations of AI techniques.

Example-1:Tic-Tac-Toe

Thefirstapproach (simple)

The Tic-Tac-Toe game consists of a nine element vector called BOARD; it represents the numbers 1 to 9 in three rows.



1	2	3
4	5	6
7	8	9

Anelement contains the value0 forblank, 1 for Xand 2 for O. AMOVETABLEvector consists of 19,683 elements (3^9) and is needed where each element is a nine element vector. The contents of the vector are especially chosen to help the algorithm.

Thealgorithm makesmoves bypursuingthe following:

1. Viewthevector as aternarynumber.Convert it toa decimal number.
2. UsethedecimalnumberasanindexinMOVETABLEandaccessthe vector.
3. Set BOARD to this vector indicating how the board looks after the move. This approach is capableintimebutithasseveraldisadvantages.Ittakesmorespaceandrequiresstunning

effort to calculate the decimal numbers. This method is specific to this game and cannot becompleted.

Thesecondapproach

The structure of the data is as before but we use 2 for a blank, 3 for an X and 5 for an O. A variable called TURN indicates 1 for the first move and 9 for the last. The algorithm consists of three actions:

MAKE2 which returns 5 if the centre square is blank; otherwise it returns any blank non-corner square, i.e. 2, 4, 6 or 8. POSSWIN (p) returns 0 if player p cannot win on the next move and otherwise returns the number of the square that gives a winning move.

It checks each line using products 3*3*2 = 18 gives a win for X, 5*5*2=50 gives a win for O, and the winning move is the holder of the blank. GO (n) makes a move to square n setting BOARD[n] to 3 or 5.

This algorithm is more involved and takes longer but it is more efficient in storage which compensates for its longer time. It depends on the programmer's skill.

Thefinal approach

The structure of the data consists of BOARD which contains a nine element vector, a list of board positionsthat couldresult from the next moveand a number representingan estimation ofhow the board position leads to an ultimate win for the player to move.

This algorithm looks ahead to make a decision on the next move by deciding which the most promising move or the most suitable move at any stage would be and selects the same.

Consider all possible moves and replies that the program can make. Continue this process for as long as time permits until a winner emerges, and then choose the move that leads to the computer program winning, if possible in the shortest time.

Actuallythisismost difficultto program bya good limit but it isasfar thatthe techniquecan be extended to in any game. This method makes relatively fewer loads on the programmer in termsof the game technique but the overall game strategy must be known to the adviser.

Example-2:QuestionAnswering

Let us consider Question Answering systems that accept input in English and provide answers also in English. This problem is harder than the previous one as it is more difficult to specify the problem properly. Another area of difficulty concerns deciding whether the answer obtained is correct, or not, and further what is meant by 'correct'. For example, consider the following situation:

Text

Rani went shopping for a new Coat. She found a red one she really liked.
Whenshegothome,shefoundthatitwentperfectlywithherfavouritedress.

Question

1. WhatdidRani goshoppingfor?

2. Whatdid Rani findthat she liked?
3. Did Rani buyanything?

Method1

Data Structures

A set of templates that match common questions and produce patterns used to match against inputs. Templates and patterns are used so that a template that matches agiven question is associated with the corresponding patternto findtheanswer in the input text. Forexample, the template who did **x y** generates **x y z** if a match occurs and **z** is the answer to the question. The given text and the question are both stored as strings.

Algorithm

Answeringaquestion requiresthe followingfoursteps to be followed:

- Comparethetemplateagainstthequestionsandstoreallsuccessfulmatches toproducea set of text patterns.

- Passthesetextpatternsthroughasubstitutionprocesstochangetheperson orvoiceand produce an expanded set of text patterns.

- Applyeachofthesepatternstothetext;collectalltheanswersandthenprint the answers.

Example

In**question1**weusethetemplateWHATDIDXYwhichgeneratesRanigoshoppingfor**z**and after substitution we get Rani goes shopping for **z** and Rani went shopping for **z** giving **z** [equivalence] a new coat

In**question2**weneedaverylargenumberoftemplatesandalsoaschemetoallowtheinsertion of ‘find’ before ‘that she liked’; the insertion of ‘really’ in the text; and the substitution of ‘she’ for ‘Rani’ gives the answer ‘a red one’.

Question3cannotbeanswered.

Comments

This is a very primitive approach basically not matching the criteria we set for intelligenceandworsethanthat,usedinthegame.Surprisinglythistypeoftechnique was actually used in ELIZA which will be considered later in the course.

Method2

Data Structures

A structure called English consists of a dictionary, grammar and some semantics about the vocabulary we are likely to come across. This data structure provides the knowledge to convert English text into a storable internal form and also to convert the response back into English. The structured representation of the text is a processed form and defines the context of the input text by making explicit all references such as pronouns. There are three types of such *knowledge representation* systems: production rules of the form ‘if x then y’, slot and filler systems and statements in mathematical logic. The system used here will be the slot and filler system.

Take,forexamplesentence:

‘Shefoundaredoneshereallyliked’.

Event2		**Event2**	
instance:	finding	instance:	liking
tense:	past	tense:	past
agent:	Rani	modifier:	much
object:	Thing1	object:	Thing1

Thing1	
instance:	coat
colour:	red

Thequestion isstored intwo forms:as inputand intheaboveform.

Algorithm

- Convert the question to a structured form using English know how, then use a marker to indicatethesubstring(like‘who’ or‘what’) ofthe structure, that should be returned as an answer. If aslot and fillersystem is used aspecial marker can beplaced in morethanone slot.
- Theanswerappears bymatchingthisstructured formagainst thestructuredtext.
- The structured form is matched against the text and the requested segments of the question are returned.

Examples

Bothquestions1 and2generateanswersviaanewcoatandaredcoat respectively. Question3cannotbeanswered,becausethereisnodirectresponse.

Comments

This approach is more meaningful than the previous one and so is more effective. The extra powergivenmustbepaidforbyadditionalsearchtimeintheknowledgebases.Awarning must be given here: that is – to generate unambiguous English knowledge base is a complex task and must be left until later in the course. The problems of handling pronouns are difficult.

Forexample:

Raniwalkeduptothesalesperson:sheaskedwherethetoydepartmentwas. Rani walked up to the salesperson: she asked her if she needed any help.

Whereasintheoriginaltextthelinkageof‘she’to‘Rani’iseasy,linkageof ‘she’ineachofthe abovesentencestoRani andtothesalesperson requiresadditional knowledge aboutthecontext via the people in a shop.

Method3

Data Structures

World model contains knowledge about objects, actions and situations that are described in the input text. This structure is used to create integrated text from input text. The diagram shows how the system's knowledge of shopping might be represented and stored. This information is known as a script and in this case is a shoppingscript. (**See figure 1.1 next page)**

1.8.2.12 Algorithm

Convert the question to a structured form using both the knowledge contained in Method 2 and the World model, generating even more possible structures, since even more knowledge is being used. Sometimes filters are introduced to prune the possible answers.

To answer a question, the scheme followed is: Convert the question to a structured form as before but use the world model to resolve any ambiguities that may occur. The structuredform is matched against the text and the requested segments of the question are returned.

Example

Bothquestions1and2generateanswers,asinthe previousprogram.Question3cannow be answered. The shopping script is instantiated and from the last sentence the path through step 14 is the one used to form the representation. ‘M’ is bound to the red coat-got home. ‘**Rani buys a red coat’** comes from step 10 and the integrated text generates that she bought a red coat.

Comments

This program is more powerful than both the previous programs because it has more knowledge. Thus, like the last game program it is exploiting AItechniques. However, we are not yet in a position to handle any English question. The major omission is that of a general reasoning mechanism known as inference to be used when the required answer is not explicitly given in the input text. But this approach can handle, with some modifications, questions of the following form with the answer—Saturday morning Rani went shopping. Her brother tried tocall her but she did not answer.

Question: Whycouldn’tRani’sbrotherreachher?

Answer: Becauseshewasnotin.

This answer is derived because we have supplied an additional fact that a person cannot be in two places at once. This patch is notsufficiently general so as to work in all cases and does not provide the type of solution we are really looking for.

Shopping Script: C - Customer, S - Salesperson

Props: M - Merchandize, D - Money-dollars, Location: L - a Store.

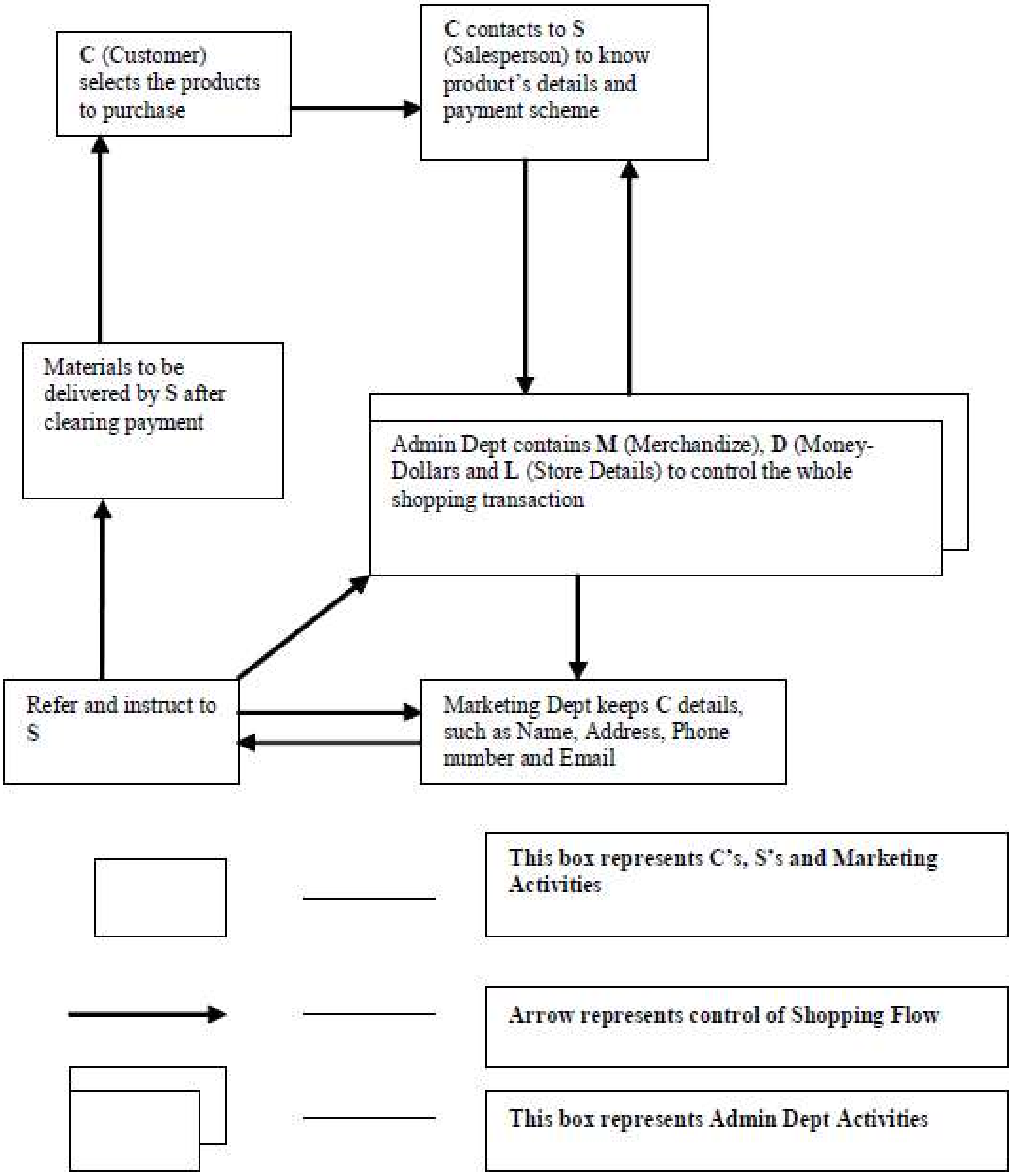

C (Customer) selects the products to purchase
C contacts to S (Salesperson) to know product's details and payment scheme
Materials to be delivered by S after clearing payment
Admin Dept contains M (Merchandize), D (Money-Dollars and L (Store Details) to control the whole shopping transaction
Refer and instruct to S
Marketing Dept keeps C details, such as Name, Address, Phone number and Email
This box represents C's, S's and Marketing Activities
Arrow represents control of Shopping Flow
This box represents Admin Dept Activities

Fig. 1.1 Diagrammatic Representation of Shopping Script

EVELOFTHEAI MODEL

‘Whatisourgoalintryingtoproduceprogramsthatdotheintelligentthingsthatpeople do?’

Arewetryingto produceprogramsthat dothetasks thesamewaythat peopledo?
OR
Arewetryingtoproduceprogramsthatsimplydothetaskstheeasiestwaythatis possible?

Programs in the first class attempt to solve problems that a computer can easilysolve and do not usually use AI techniques. AI techniques usually include a search, as no direct method is available,theuseofknowledgeabouttheobjectsinvolvedintheproblemareaandabstractionon which allows an element of pruning to occur, and to enable a solution to be found in real time; otherwise, the data could explode in size. Examples of these trivial problems in the first class, which are now of interest only to psychologists are EPAM (Elementary Perceiver and Memorizer) which memorized garbage syllables.

Thesecondclassofproblemsattemptstosolveproblemsthatarenon-trivialforacomputerand use AI techniques. We wish to model human performance on these:

1. Totestpsychologicaltheoriesofhumanperformance.Ex.PARRY[Colby,1975] –a program to simulate the conversational behavior of a paranoid person.
2. Toenablecomputerstounderstandhumanreasoning–forexample,programsthat answer questions based upon newspaper articles indicating human behavior.
3. Toenablepeopletounderstandcomputerreasoning.Somepeoplearereluctanttoaccept computer results unless they understand the mechanisms involved in arriving at the results.
4. Toexploittheknowledgegainedbypeoplewhoarebestatgatheringinformation.This persuaded the earlier workers to simulate human behavior in the SB part of AISB simulated behavior. Examples of this type of approach led to GPS (General Problem Solver).

Questions for Practice:

1. Whatis*intelligence*?Howdowemeasureit?Arethesemeasurementsuseful?
2. When the temperature falls and the thermostat turns the heater on, does it act because it *believes*theroomtobetoocold? Doesit*feel*cold?Whatsortsofthingscanhavebeliefs or feelings? Is this related to the idea of consciousness?
3. Some people believe that the relationship between your mind (a non-physical thing) and yourbrain(thephysicalthinginside yourskull)isexactlyliketherelationshipbetweena computational process (a non-physical thing) and a physical computer. Do you agree?
4. Howgoodaremachinesatplayingchess? Ifamachinecanconsistentlybeatallthebest human chess players, does this prove that the machine is *intelligent*?
5. Whatis AITechnique?ExplainTic-Tac-ToeProblemusingAITechnique.

PROBLEMS,PROBLEMSPACESANDSEARCH

Tosolvetheproblemof buildingasystemyoushouldtakethefollowingsteps:

1. Definetheproblemaccuratelyincludingdetailedspecificationsandwhatconstitutesa suitable solution.
2. Scrutinizetheproblemcarefully,forsomefeaturesmayhaveacentralaffectonthe chosen method of solution.
3. Segregateandrepresentthebackgroundknowledgeneededinthesolutionofthe problem.
4. Choosethe best solvingtechniques for theproblem to solve a solution.

***Problemsolving*isaprocess**of generatingsolutionsfromobserveddata.

- a*'problem'*is characterized byaset of*goals*,
- asetof*objects*, and
- aset of*operations*.

Thesecould be ill-defined andmayevolve duringproblem solving.

- A**'*problem space*'**isanabstractspace.
  - ✓ A problem space encompasses all *valid states* that can be generated by the applicationofanycombinationof*operators*onanycombinationof*objects*.
  - ✓ Theproblemspacemaycontainoneormore*solutions*.Asolutionisa combination of *operations* and *objects* that achieve the *goals*.
- A'***search***'referstothe searchforasolutionina problem space.
  - ✓ Searchproceedswithdifferenttypesof'*searchcontrol strategies*'.
  - ✓ The*depth-firstsearchandbreadth-first search*arethetwocommon*search strategies.*

AI-GeneralProblemSolving

Problemsolving hasbeen thekeyareaofconcern forArtificialIntelligence.

Problemsolvingisaprocessofgeneratingsolutionsfromobservedorgivendata.Itishowever not always possible to use direct methods (i.e. go directly from data to solution). Instead, problem solving often needs to use indirect or modelbased methods.

General Problem Solver(GPS) was a computer program created in 1957 bySimon and Newell tobuildauniversalproblemsolvermachine. *GPS* wasbasedonSimonand Newell'stheoretical work on logic machines. *GPS* in principle can solve any formalized symbolic problem, such as theorems proof and geometric problems and chess playing. *GPS* solved manysimple problems, suchastheTowersofHanoi,thatcouldbesufficientlyformalized,but ***GPScouldnotsolveany real-world problems***.

Tobuildasystemtosolveaparticularproblem, weneed to:

- Definetheproblemprecisely–findinputsituationsaswellasfinalsituationsforan acceptable solution to the problem

- Analyzetheproblem–findfewimportantfeaturesthatmayhaveimpactonthe appropriateness of various possible techniques for solving the problem
- Isolateand represent taskknowledgenecessarytosolve theproblem
- Choosethe best problem-solvingtechnique(s)andapplyto theparticular problem

Problemdefinitions

Aproblemisdefinedbyits‘*elements*’andtheir‘*relations*’.Toprovideaformaldescriptionofa problem, we need to do the following:

a. Definea*statespace*thatcontainsallthepossibleconfigurationsoftherelevantobjects, including some impossible ones.
b. Specifyoneormorestatesthatdescribepossiblesituations,fromwhichtheproblem-solving process may start. These states are called *initial states*.
c. Specifyoneor morestatesthat would be acceptablesolution to the problem.

Thesestates arecalled*goal states*.

Specifyaset of*rules* thatdescribetheactions (*operators*) available.

Theproblemcanthenbesolvedbyusingthe*rules*,incombinationwithanappropriate*control strategy*, to movethrough the *problem space*until a*path* from an *initial state* to a *goal state*is found. This process is known as ***‘search’***. Thus:

- *Search*isfundamentalto theproblem-solvingprocess.
- *Search*isageneralmechanismthatcanbeusedwhenamoredirect method is notknown.
- *Search* provides the framework into which more direct methods for solving subparts of aproblemcanbeembedded.Averylargenumberof AI problems are formulated as search problems.
- Problem space

A*problemspace*isrepresented byadirected graph,where*nodes*representsearchstateand*paths* representtheoperatorsappliedtochangethe*state*.

To simplifysearch algorithms, it is often convenient to logicallyand programmaticallyrepresent aproblemspaceasa **tree**.A*tree*usuallydecreasesthecomplexityofasearchatacost.Here,the cost is due to duplicating some nodes on the tree that were linked numerous times in the graph, e.g.node***B*** andnode***D.***

A*treeisagraph*inwhichanytwovertices areconnectedbyexactlyone path.Alternatively,any connected *graph with no cycles is a tree*.

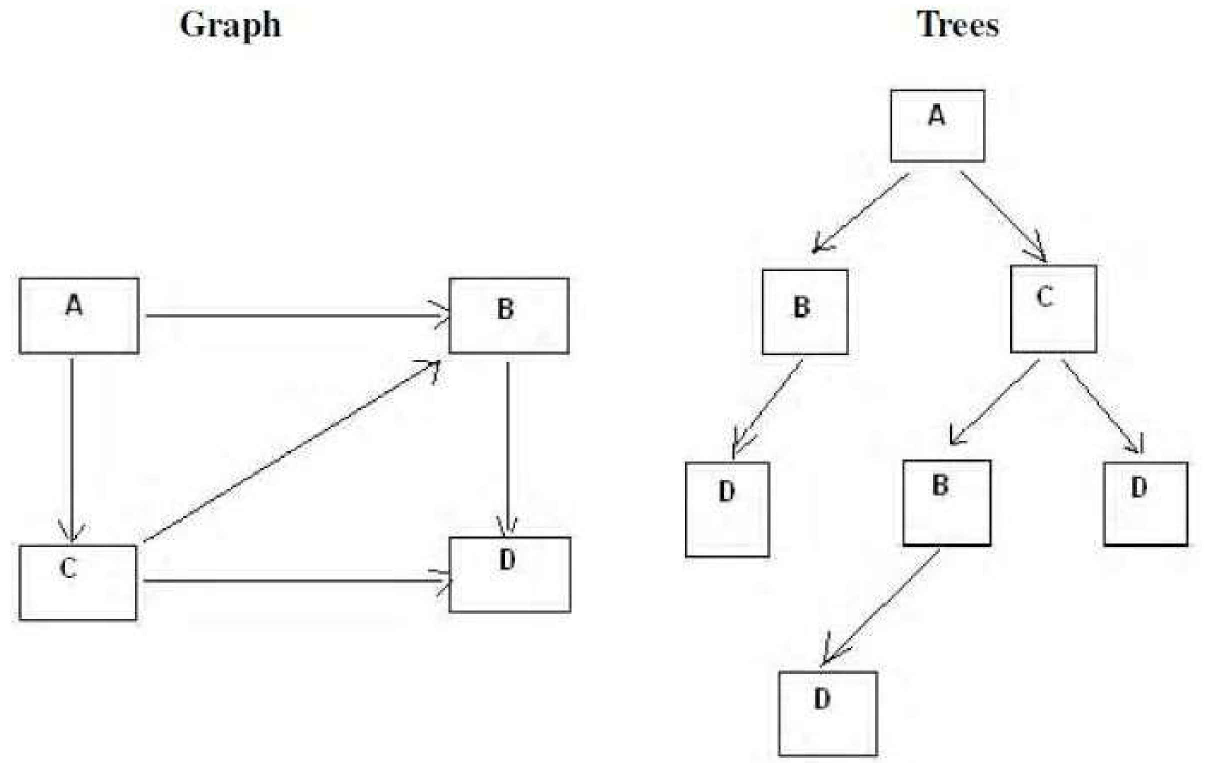


Fig. 2.1 Graph and Tree

- **Problemsolving:**Theterm,ProblemSolvingrelatestoanalysisinAI.Problemsolvingmaybe characterizedasasystematicsearchthrougharangeofpossibleactionsto reachsomepredefined goal or solution. Problem-solving methods are categorized as *special purpose* and *general purpose*.

- A*special-purposemethod*istailor-madeforaparticularproblem,oftenexploitsveryspecific features of the situation in which the problem is embedded.

- A*general-purposemethod*isapplicabletoawidevarietyofproblems.OneGeneral-purpose technique used in AI is *‘means-end analysis’:* a step-bystep, or incremental, reduction of the difference between current state and final goal.

DEFININGPROBLEMASASTATESPACE SEARCH

Tosolvetheproblemofplayingagame,werequiretherulesofthegameand targetsforwinning as well as representing positions in the game. The opening position can be defined as the initial state and a winningposition as a goal state. Moves from initial state to otherstates leadingto the goal state follow legally. However, the rules are far too abundant in most games— especially in chess, where they exceed the number of particles in the universe. Thus, the rules cannot be supplied accurately and computer programs cannot handle easily. The storage also presents another problem but searching can be achieved by hashing.

The number of rules that are used must be minimized and the set can be created by expressing eachruleinaformaspossible.Therepresentationofgamesleadstoastatespacerepresentation and it is common for well-organized games with some structure. This representation allows for theformaldefinitionofa problem thatneedsthe movementfromasetofinitialpositionsto one of a set of target positions. It means that the solution involves using known techniques and a systematic search. This is quite a common method in Artificial Intelligence.

StateSpace Search

A*statespace*representsaproblemintermsof*states*and*operators*thatchangestates. A state space consists of:

- Arepresentationofthe*states*thesystemcanbein.Forexample,ina board game, the board represents the current state of the game.
- A set of *operators* that can change one state into another state. In a board game, the operators are the legal moves from any given state. Often the operatorsarerepresentedasprogramsthatchangeastaterepresentationto represent the new state.
- An *initial state*.
- A set of *final states*; some of these maybe desirable, others undesirable. Thissetisoftenrepresentedimplicitlybyaprogramthatdetectsterminal states.

TheWaterJugProblem

Inthisproblem,weusetwojugscalled **four**and**three;**fourholdsamaximumoffourgallonsof water and **three** a maximum of three gallons of water. How can we get two gallons of water in the **four** jug?

Thestatespaceisasetofprearrangedpairsgivingthenumberofgallonsof waterinthepairof jugs at any time, i.e., (**four, three**) where **fou**r = 0, 1, 2, 3 or 4 and **three** = 0, 1, 2 or 3.

Thestartstateis(0,0)andthegoalstateis(2,n)wherenmaybeanybutitis limitedto**three** holdingfrom 0 to 3 gallons of water or empty. Three and four shows the name and numerical number shows the amount of water in jugs for solving the water jug problem. The major production rules for solving this problem are shown below:

Initial condition	**Goalcomment**
1. (four,three) iffour <4	(4,three)fillfourfromtap
2. (four,three)ifthree<3	(four,3) fillthreefromtap
3. (four,three) Iffour>0	(0,three)emptyfourintodrain
4. (four,three)ifthree>0	(four,0)emptythreeintodrain
5. (four, three) if four + three<4	(four+three,0)emptythreeinto four
6. (four, three) if four + three<3	(0,four+three)emptyfourinto three
7. (0,three)Ifthree>0	(three, 0)emptythreeintofour
8. (four,0)iffour>0	(0,four) emptyfourinto three
9. (0,2)	(2, 0)emptythreeintofour
10. (2,0)	(0,2)emptyfourintothree
11. (four, three) if four < 4	(4,three-diff)pourdiff,4-four,into four from three
12. (three, four) if three < 3	(four-diff,3)pourdiff,3-three,into three from four and a solution is given below four three rule

***(Fig.2.2** ProductionRulesforthe WaterJugProblem)*

GallonsinFourJug	**Gallonsin ThreeJug**	**RulesApplied**
0	0	-
0	3	2
3	0	7
3	3	2
4	2	11
0	2	3
2	0	10

***(Fig.2.3** OneSolutiontothe WaterJug Problem)*

The problem solved by using the production rules in combination with an appropriate control strategy, moving through the problem space until a path from an initial state to a goal state is found. In this problem solving process, search is the fundamental concept. For simple problemsit is easier to achieve this goal by hand but there will be cases where this is far too difficult.

PRODUCTIONSYSTEMS

Production systems provide appropriate structures for performing and describing search processes. A production system has four basic components as enumerated below.

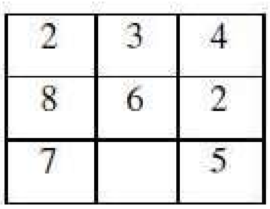
- A set of rules each consisting of a left side that determines the applicability of the rule and a right side that describes the operation to be performed if the rule is applied.
- Adatabaseofcurrentfactsestablishedduringthe processof inference.

- A control strategy that specifies the order in which the rules will be compared with facts in the database and also specifies how to resolve conflicts in selection of several rules or selection of more facts.
- Arulefiringmodule.

The production rules operate on the knowledge database. Each rule has a precondition—that is, either satisfied or not by the knowledge database. If the precondition is satisfied, the rule can be applied. Application of the rule changes the knowledge database. The control system chooses whichapplicableruleshould beappliedandceasescomputationwhenaterminationconditionon the knowledge database is satisfied.

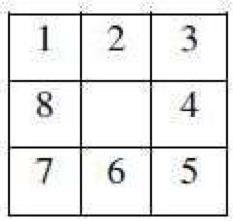
Example:Eightpuzzle (8-Puzzle)

The 8-puzzle is a 3 × 3 arraycontaining eight square pieces, numbered 1 through 8, and oneemptyspace. Apiececanbemovedhorizontallyorverticallyintotheemptyspace,ineffect exchanging the positions of the piece and the empty space. There are four possible moves, UP (move the blank space up), DOWN, LEFT and RIGHT. The aim of the game is to make a sequence of moves that will convert the board from the start state into the goal state:



2	3	4
8	6	2
7		5

Initial State



1	2	3
8		4
7	6	5

Goal State

Thisexamplecan besolvedbytheoperator sequenceUP,RIGHT, UP,LEFT, DOWN.

Example:Missionariesand Cannibals

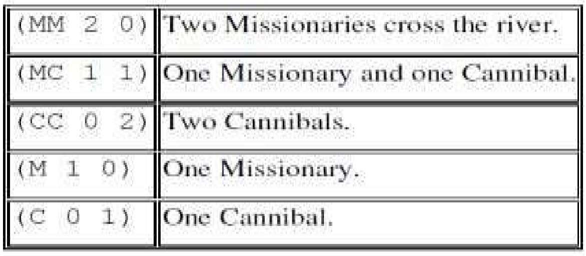
TheMissionariesandCannibalsproblemillustratestheuseofstatespacesearchfor planning under constraints:

Three missionaries and three cannibals wish to cross a river using a two person boat. If atanytimethecannibalsoutnumberthemissionariesoneithersideoftheriver,theywilleatthe missionaries. How can a sequence of boat trips be performed that will get everyone to the other side of the river without any missionaries being eaten?

Staterepresentation:

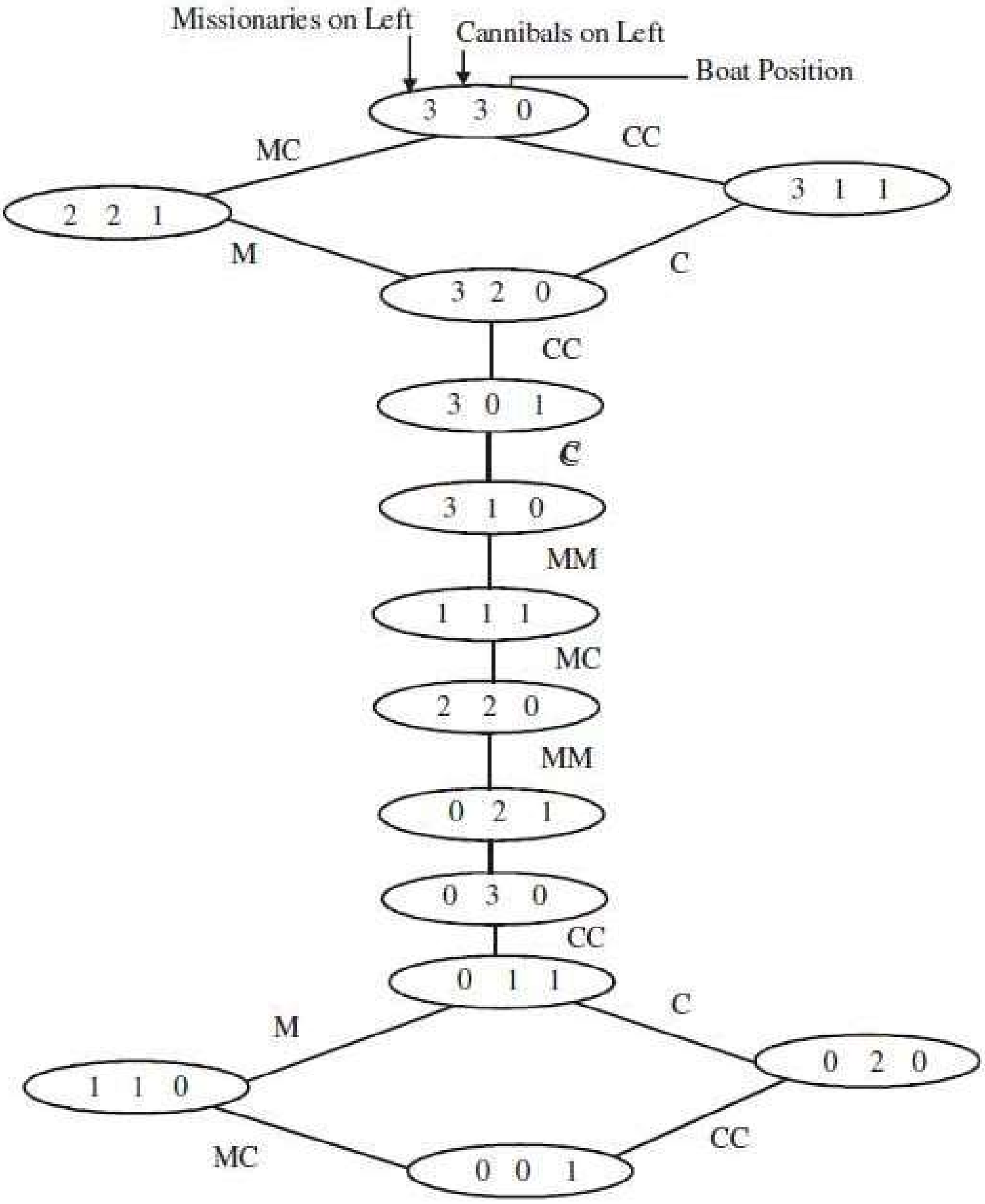
1. BOATposition:original(T)orfinal(NIL) side ofthe river.
2. NumberofMissionariesand Cannibalson the original sideof theriver.
3. Startis (T 33); Goalis (NIL0 0).

Operators:



(MM 2 0)	Two Missionaries cross the river.
(MC 1 1)	One Missionary and one Cannibal.
(CC 0 2)	Two Cannibals.
(M 1 0)	One Missionary.
(C 0 1)	One Cannibal.

Missionaries/Cannibals Search Graph



Control Strategies

Theword '***search***'refers tothesearchforasolution ina *problem space*.

- Searchproceedswithdifferenttypesof *'**searchcontrolstrategies**'*.
- Astrategyisdefined bypickingthe order in which the nodesexpand.

The Search strategies are evaluated along the following dimensions: Completeness, Time complexity,Spacecomplexity,Optimality(thesearch-relatedtermsarefirstexplained,andthen the search algorithms and control strategies are illustrated next).

Search-relatedterms

- **Algorithm'sperformanceandcomplexity**

Ideallywewantacommonmeasuresothatwecancompareapproachesin ordertoselect the most appropriate algorithm for a given situation.

- ✓ *Performance*ofanalgorithmdependsoninternalandexternalfactors.

Internalfactors/Externalfactors

- ***Time***requiredto run
- ***Size*** of input to the algorithm
- ***Space***(memory) requiredtorun
- ***Speed***ofthe computer
- ***Quality*** of the compiler

- ✓ *Complexity*isameasure oftheperformanceofanalgorithm.*Complexity* measuresthe internal factors,usuallyin timethanspace.

- **Computationalcomplexity\**

Itis themeasureof resources interms of ***Time***and ***Space***.

- ✓ If***A***isanalgorithmthatsolvesadecisionproblem***f***,thenrun-timeof***A***isthenumberof steps taken on the input of length ***n.***
- ✓ *TimeComplexity**T(n)*** of adecision problem***f***is therun-timeofthe 'best'algorithm***A*** for***f.***
- ✓ *SpaceComplexity**S(n)***ofadecisionproblem***f***istheamountofmemoryusedbythe 'best' algorithm ***A*** for***f.***

- '**Big-O**'**notation**

The***Big-O***, theoretical*measureof the executionof an*algorithm, usuallyindicates the***time***or the ***memory***needed, giventhe problem size***n***, whichis usuallythenumber of items.

- ***Big-O***notation

The***Big-O***notation isusedto give anapproximation tothe*run-time-efficiencyofan algorithm;* theletter'**O**'isfororder ofmagnitudeofoperationsorspace atrun-time.

- The***Big-O***ofan Algorithm***A***
  - ✓ Ifanalgorithm***A***requirestimeproportionalto***f(n)***,thenalgorithm***A***issaidtobe of order ***f(n)***, and it is denoted as ***O(f(n))***.
  - ✓ Ifalgorithm***A***requirestimeproportionalto***n2***,thentheorderofthealgorithmis said to be ***O(n2).***
  - ✓ Ifalgorithm***A***requirestimeproportionalto***n***,thentheorderofthealgorithmis said to be ***O(n).***

Thefunction ***f(n)***iscalled the algorithm's *growth-rate function*. In other words, if an algorithm hasperformancecomplexity***O(n)***,thismeansthattherun-time***t***shouldbedirectlyproportional to ***n***, ie ***t • n or t = k n*** where ***k*** is constant of proportionality.
Similarly,foralgorithmshavingperformancecomplexity***O(log2(n)),O(logN),O(NlogN), O(2N)*** and so on.

Example1:
Determinethe***Big-O***ofan algorithm:

Calculatethesum ofthe***n*** elements inan integer array***a[0..n-1]***.

Line no.	**Instructions**	**Noofexecutionsteps**
line1	sum	1
line2	for(i =0; i <n; i++)	n +1
line3	sum +=a[i]	n
line4	print sum	1
	Total	**2n + 3**

Thus,thepolynomial***(2n +3)***isdominatedbythe1sttermas***n***whilethenumberofelementsin the array becomes very large.

- Indeterminingthe***Big-O***,ignoreconstantssuchas ***2*** and***3***.Sothe algorithm isoforder***n***.
- So the***Big-O***of thealgorithm is***O(n)***.
- Inotherwordstherun-timeofthisalgorithmincreasesroughlyasthesizeoftheinputdata***n***, e.g., an array of size ***n***.

Treestructure
Treeisa wayof organizingobjects, relatedinahierarchical fashion.

- Treeisatypeofdatastructureinwhicheach*element*isattachedtooneormore elements directly beneath it.
- Theconnectionsbetweenelementsarecalled*branches*.
- Treeisoftencalled *inverted trees*becauseitis drawnwith the *root* atthetop.
- Theelementsthathavenoelementsbelow themarecalled*leaves*.
- A*binarytree*isa specialtype:eachelementhasonlytwo branchesbelow it.

Properties

- Treeisaspecial caseofa *graph*.
- Thetopmost nodein a treeiscalledthe *root node*.
- Atrootnodealloperationson thetreebegin.
- Anodehasatmost oneparent.
- Thetopmostnode(rootnode)hasno parents.
- Eachnodehaszeroor more*childnodes*, whicharebelowit .
- The nodes at the bottommost level of the tree are called *leaf nodes*.
Since*leafnodes*areatthebottommostlevel,theydonothavechildren.
- Anodethathasa child iscalledthechild's *parent node*.
- The*depth of a node* ***n*** is thelengthof thepath from theroot to thenode.
- Theroot nodeisat depth zero.

- **Stacksand Queues**

The*Stacks*and*Queues*aredatastructuresthatmaintaintheorderof*last-in,first-out*and*first-in, first-out* respectively. Both *stacks* and *queues* are often implemented as linked lists, but that is not the only possible implementation.

Stack-Last InFirstOut (LIFO)lists

- Anordered list;asequenceof items,piled oneontopof theother.
- The***insertions***and***deletions***aremade atoneendonly,called***Top***.
- IfStack***S=(a[1],a[2],............. a[n])***then***a[1]***is bottom mostelement
- Anyintermediate element***(a[i])***is on top ofelement***a[i-1], 1< i <= n.***
- InStackalloperationtakeplaceon***Top***.

The***Pop***operationremovesitemfromtopofthestack. The ***Push*** operation adds an item on top of the stack.

Queue-FirstInFirstOut(FIFO)lists

- Anorderedlist;asequenceofitems;therearerestrictionsabouthowitems canbeaddedtoand removed from the list. A queue has two ends.
- All***insertions***(enqueue) takeplaceatoneend,called***Rear***or***Back***
- All***deletions***(dequeue) takeplaceat other end,called***Front***.
- IfQueuehas ***a[n]*** asrear elementthen ***a[i+1]***isbehind ***a[i],1< i<= n.***
- Alloperation takesplaceat one end ofqueueor theother.

The***Dequeue***operationremovestheitemat***Front***ofthequeue. The ***Enqueue*** operation adds an item to the ***Rear*** of the queue. **Search**

*Search*isthesystematicexaminationof *states*tofindpathfromthe*start/rootstate*tothe*goal state*.

- Searchusuallyresultsfrom alack of knowledge.
- Searchexploresknowledgealternativesto arriveatthebestanswer.
- Searchalgorithmoutputisasolution,thatis,apathfromtheinitialstatetoastatethatsatisfies the goal test.

Forgeneral-purposeproblem-solving–'***Search'isanapproach***.

- Searchdealswithfinding*nodes*havingcertainpropertiesina*graph*thatrepresentssearch space.
- Searchmethodsexplorethesearchspace'intelligently',evaluatingpossibilitieswithout investigating every single possibility.

Examples:

- ForaRobotthismightconsistofPICKUP,PUTDOWN,MOVEFORWARD,MOVEBACK, MOVELEFT, and MOVERIGHT—until the goal is reached.
- PuzzlesandGameshaveexplicitrules:e.g.,the '***TowerofHanoi***' puzzle

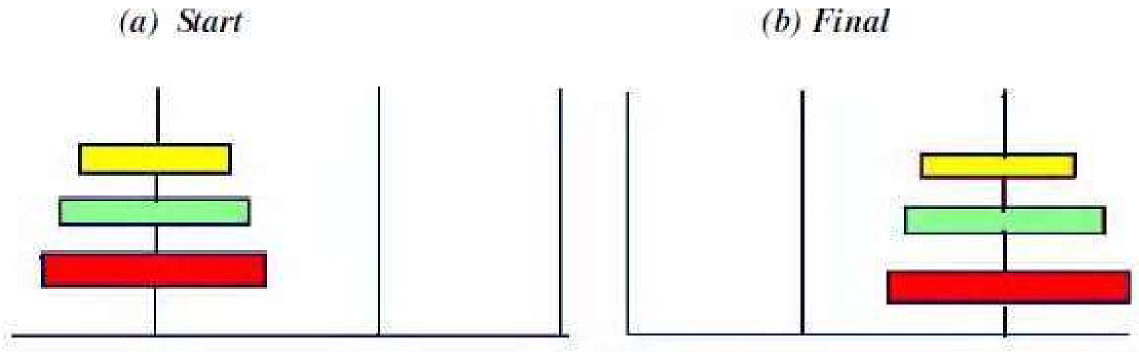


Fig. 2.4 Tower of Hanoi Puzzle

Thispuzzle involvesasetof ringsof differentsizes thatcan beplaced onthreedifferent pegs.
- ThepuzzlestartswiththeringsarrangedasshowninFigure2.4(a)
- Thegoal ofthispuzzleistomove themall asto Figure 2.4(b)
- Condition:Onlythetop ringona pegcanbemoved,anditmayonlybe placedonasmaller ring, or on an empty peg.

Inthis*TowerofHanoi*puzzle:
- Situationsencountered whilesolvingtheproblemaredescribedas*states*.
- Setofallpossibleconfigurationsofringsonthe pegsiscalled*'problem space*'.
- **States**

A*State*isarepresentationofelementsinagivenmoment. A problem is defined by its *elements* and their *relations*.
Ateachinstant of aproblem,theelementshavespecificdescriptorsandrelations;the*descriptors* indicatehowtoselectelements?
Amongall possible states,therearetwospecial states called:
- ✓ *Initialstate* – the start point
- ✓ *Finalstate*–thegoal state

- **StateChange:**SuccessorFunction

A'*successorfunction*'isneededforstatechange.TheSuccessorFunction movesonestateto another state.
Successor Function:
- ✓ Itisadescriptionof possibleactions;asetof operators.
- ✓ Itisatransformationfunctiononastaterepresentation,whichconvertsthatstateinto another state.
- ✓ Itdefinesarelationof accessibilityamongstates.
- ✓ Itrepresentstheconditionsofapplicabilityofastateandcorrespondingtransformation function.

- **Statespace**

A*statespace*is thesetofall*states*reachable from the*initial state*.
- ✓ A*statespace*forms a*graph*(ormap)inwhichthe*nodes*arestatesandthe *arcs*between nodes are actions.
- ✓ Ina*statespace*,a*path* isa sequenceofstatesconnected byasequenceofactions.
- ✓ The*solution* of aproblem is part ofthe map formed bythe*state space*.

• **Structureofastatespace**

The*structures*of a*statespace*are*trees*and *graphs*.

- ✓ A*tree*isahierarchicalstructureina graphicalform.
- ✓ A*graph*isanon-hierarchical structure.

• A*tree*has onlyone path to agivennode;
i.e.,a*tree*has one andonlyonepathfromanypoint toanyotherpoint.

• A *graph* consists of a set of nodes (vertices) and a set of edges (arcs). Arcs establish relationships(connections)betweenthenodes;i.e.,agraphhasseveralpathstoagivennode.

• The*Operators*aredirected*arcs*between nodes.

A*search*processexploresthe *statespace*.Inthe worstcase,thesearchexploresallpossible *paths*between the*initial state* and the*goal state*.

• **Problemsolution**

Inthe*statespace*, a*solution* is a path from the*initial state* to a*goalstate*or,sometimes, just a *goal state*.

- ✓ A*solutioncostfunction*assignsanumericcosttoeach*path*;italsogivesthecostof applying the *operators* to the *states*.
- ✓ A*solutionquality*ismeasuredbythe*pathcostfunction*;andanoptimalsolutionhasthe lowest path cost among all solutions.
- ✓ The*solutions*can be*any or optimal or all*.
- ✓ Theimportanceof cost depends on theproblemand the typeof solution asked

• **Problemdescription**

Aproblemconsistsofthedescription of:

- ✓ Thecurrentstate oftheworld,
- ✓ Theactions thatcan transform onestateof theworld into another,
- ✓ Thedesired stateof the world.

Thefollowingaction one takento describetheproblem:

- ✓ *Statespace*isdefinedexplicitlyorimplicitly

A*statespace*shoulddescribeeverythingthatisneededtosolveaproblem andnothingthatis not needed to solve the problem.

- ✓ *Initialstate* isstart state
- ✓ *Goal state* is the conditions it has to fulfill.

Thedescriptionbyadesiredstatemaybecompleteorpartial.

- ✓ *Operators*aretochange state
- ✓ Operatorsdoactionsthatcantransformonestateinto another;
- ✓ Operators consistof: Preconditionsand Instructions;

Preconditionsprovidepartialdescriptionofthestateoftheworldthatmustbetrueinorderto perform the action, and

Instructions telltheuserhowtocreatethenext state.

- Operatorsshouldbe asgeneralaspossible,soastoreducetheirnumber.
- *Elementsofthe domain* hasrelevancetothe problem
  - ✓ Knowledgeofthestartingpoint.
- *Problemsolving* is findingasolution
  - ✓ Findanorderedsequenceofoperatorsthattransformthecurrent(start)state into a goal state.

- *Restrictions* aresolution qualityany, optimal,orall
    - ✓ Findingtheshortestsequence, or
    - ✓ findingthe least expensive sequencedefining cost, or
    - ✓ findinganysequenceasquicklyaspossible.

Thiscanalsobeexplainedwiththehelpofalgebraicfunctionasgivenbelow.

PROBLEMCHARACTERISTICS

Heuristics cannot be generalized, as theyaredomain specific. Production systems provideideal techniques for representing such heuristics in the form of IF-THEN rules. Most problems requiringsimulationofintelligenceuseheuristicsearch extensively.Someheuristicsareusedto define the control structure that guides the search process, as seen in the example described above.Butheuristicscan alsobeencodedintherulestorepresentthedomainknowledge.Since most AIproblems make useof knowledge and guided search through the knowledge, AIcan be described as*thestudyof techniquesforsolvingexponentiallyhardproblemsin polynomialtime by exploiting knowledge about problem domain*.

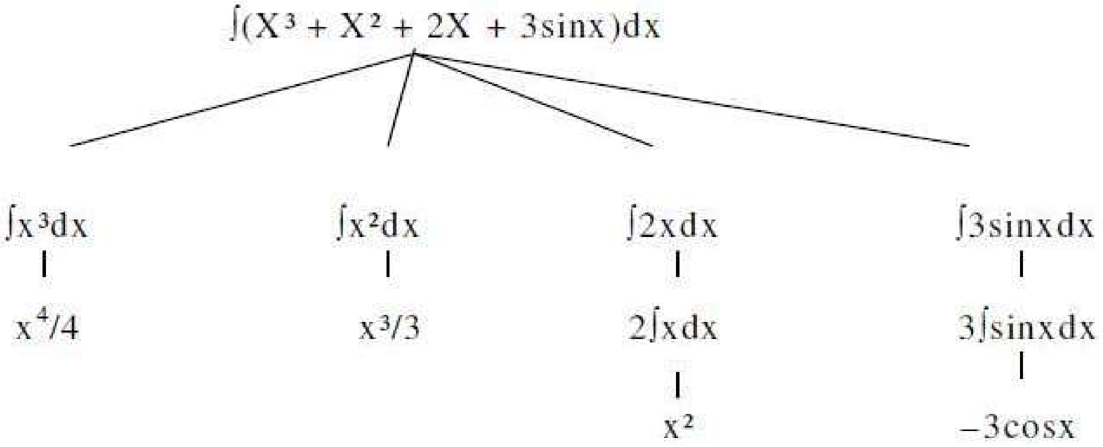
Tousetheheuristicsearchforproblemsolving,wesuggestanalysisofthe problemforthe following considerations:

- Decomposabilityof theproblem into aset ofindependent smaller subproblems
- Possibilityof undoingsolutionsteps, if theyare found to beunwise
- Predictabilityoftheproblemuniverse
- Possibilityofobtaininganobvioussolutionto aproblemwithoutcomparisonofallother possible solutions
- Typeof thesolution: whetheritis astateorapath tothegoal state
- Roleofknowledgein problem solving
- Natureofsolutionprocess: withorwithout interactingwiththeuser

The general classes of engineering problems such as planning, classification, diagnosis, monitoring and design are generally knowledge intensive and use a large amount of heuristics. Depending on the type of problem, the knowledge representation schemes and control strategies forsearch are to be adopted.Combining heuristics with thetwo basicsearch strategieshavebeen discussed above. There are a number of other general-purpose search techniques which are essentially heuristics based. Their efficiency primarily depends on how they exploit the domain-specific knowledge to abolish undesirable paths. Such search methods are called ‘weakmethods’, since the progress of the search depends heavily on the way the domain knowledge is exploited.Afewofsuchsearch techniques which form thecentreof manyAIsystems arebriefly presented in the following sections.

ProblemDecomposition

Supposetosolve theexpression is:+□(X^3+X^2 +2X +3sinx) dx



This problem can be solved bybreaking it into smaller problems, each of which we can solve by usingasmall collection ofspecificrules.Usingthistechnique ofproblemdecomposition,wecan solve very large problems very easily. This can be considered as an intelligent behaviour.

Can SolutionSteps beIgnored?

Suppose we are trying to prove a mathematical theorem: first we proceed considering that proving a lemma will be useful. Later we realize that it is not at all useful. We start with another one to prove the theorem. Here we simply ignore the first method.

Consider the 8-puzzle problem to solve: we make a wrong move and realize that mistake. But here, the control strategy must keep track of all the moves, so that we can backtrack to the initial state and start with some new move.

Consider the problem of playing chess. Here, once we make a move we never recover from that step. These problems are illustrated in the three important classes of problems mentioned below:

1. Ignorable,inwhichsolutionstepscanbeignored.Eg:TheoremProving
2. Recoverable,inwhichsolutionstepscanbeundone.Eg: 8-Puzzle
3. Irrecoverable,inwhichsolutionstepscannotbeundone.Eg:Chess

IstheProblemUniversePredictable?

Considerthe8-Puzzleproblem.Everytimewemakeamove,weknowexactly whatwillhappen. This means that it is possible to plan an entire sequence of moves and be confident what the resulting state will be. We can backtrack to earlier moves if they prove unwise.

Suppose we want to play Bridge. We need to plan before the first play, but we cannot play with certainty. So, the outcome of this game is very uncertain. In case of 8-Puzzle, the outcome isverycertain. To solve uncertain outcome problems, we follow the process of plan revision as the plan is carried out and the necessary feedback is provided. The disadvantage is that the planning in this case is often very expensive.

IsGoodSolution AbsoluteorRelative?

Consider the problem of answering questions based on a database of simple facts such as the following:

1. Sivawasaman.
2. Sivawas aworker ina company.
3. Sivawas bornin 1905.
4. Allmenare mortal.
5. Allworkersin afactorydied whentherewasanaccidentin 1952.
6. Nomortalliveslongerthan100years.

Suppose we ask a question: 'Is Siva alive?'

By representing these facts in a formal language, such as predicate logic, and then using formal inference methods we can derive an answer to this question easily.

Therearetwo waystoanswer thequestionshownbelow:

MethodI:

1. Sivawasaman.
2. Sivawas bornin 1905.
3. Allmenare mortal.
4. Nowitis2008,soSiva'sageis103years.
5. Nomortal liveslongerthan100years.

MethodII:

1. Sivais aworker inthecompany.
2. Allworkers in the companydied in 1952.

Answer:SoSivaisnotalive.Itisthe answerfromtheabove methods.

We are interested to answer the question; it does not matter which path we follow. If we follow one path successfullyto the correct answer, then there is no reason to go back and check another path to lead the solution.

CHARACTERISTICSOFPRODUCTIONSYSTEMS

Productionsystemsprovideuswithgoodwaysofdescribingtheoperationsthatcan be performed in a search for a solution to a problem.

Atthistime,two questionsmayarise:

1. Canproductionsystemsbedescribedbyasetofcharacteristics? Andhowcantheybe easily implemented?
2. Whatrelationshipsaretherebetweentheproblemtypesandthetypesofproduction systems well suited for solving the problems?

Toanswerthesequestions,firstconsiderthefollowingdefinitionsofclassesofproduction systems:

1. Amonotonicproductionsystemisaproductionsysteminwhichtheapplicationofa rule never prevents the later application of another rule that could also have been applied at the time the first rule was selected.
2. Anon-monotonicproduction systemis onein whichthis isnot true.
3. A partially communicative production system is a production system with the propertythatiftheapplicationofaparticularsequenceofrulestransformsstatePinto state Q, then any combination of those rules that is allowable also transforms state P into state Q.
4. Acommutativeproductionsystemisaproductionsystemthatisbothmonotonicand partially commutative.

Table 2.1 Four Categories of Production Systems

Production System	Monotonic	Non-monotonic
Partially Commutative	Theorem Proving	Robot Navigation
Non-partially Commutative	Chemical Synthesis	Bridge

Is there any relationship between classes of production systems and classes of problems? For any solvable problems, there exist an infinite number of production systems that show howto find solutions. Any problem that can be solved by any production system can be solved by a commutative one, but the commutative one is practically useless. It may use individual states to represent entire sequences of applications of rules of a simpler, non-commutative system. In the formalsense,thereisnorelationshipbetweenkindsofproblemsandkindsofproductionsystems Since all problems can be solved by all kinds of systems. But in the practical sense, there is definitely such a relationship between the kinds of problems and the kinds of systems that lend themselves to describing those problems.

Partially commutative, monotonic productions systems are useful for solving ignorable problems. These are important from an implementation point of view without the ability to backtrack to previous states when it is discovered that an incorrect path has been followed. Both typesofpartiallycommutativeproductionsystemsaresignificantfromanimplementationpoint; theytend to lead to many duplications of individual states during the search process. Production systems that are not partially commutative are useful for many problems in which permanent changes occur.

IssuesintheDesignofSearch Programs

Eachsearchprocesscanbeconsideredtobeatreetraversal.Theobjectofthe searchistofinda path from the initial state to a goal state using a tree. The number of nodes generated might be huge; and in practice many of the nodes would not be needed. The secret of a good search routine is to generate only those nodes that are likely to be useful, rather than having a precise tree.Therulesareusedtorepresent thetreeimplicitlyandonlytocreatenodesexplicitlyifthey are actually to be of use.

Thefollowingissuesarisewhen searching:

- Thetreecanbesearchedforwardfromtheinitialnodetothegoalstateor backwardsfromthe goal state to the initial state.
- Toselectapplicablerules,itiscriticaltohavean efficientprocedurefor matchingrulesagainst states.
- How to represent each node of the search process? This is the knowledge representation problemortheframeproblem.Ingames,anarraysuffices;in otherproblems,morecomplex data structures are needed.

Finally in terms of data structures, considering the water jug as a typical problem do we use a graphortree? Thebreadth-firststructuredoestakenoteofallnodesgeneratedbutthedepth-first one can be modified.

Checkduplicatenodes

1. Observeall nodesthatarealreadygenerated,ifanew nodeis present.
2. Ifitexists addit tothegraph.
3. Ifitalreadyexists, then

    a. Set the node that is being expanded to the point to the already existing node correspondingtoitssuccessorratherthantothenewone.Thenewonecanbethrown away.

    b. Ifthebestorshortestpathisbeingdetermined,checktoseeifthispathis betteror worse than the old one. If worse, do nothing.

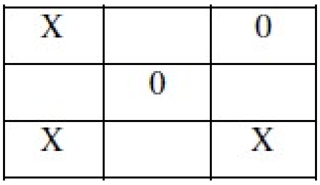
Bettersavethenewpathandworkthechangeinlengththroughthechainofsuccessornodesif necessary.

Example:Tic-Tac-Toe

State spaces are good representations for board games such as Tic-Tac-Toe. The position of a gamecanbeexplainedbythecontentsoftheboardandtheplayer whoseturnisnext.Theboard can be represented as an array of 9 cells, each of which may contain an X or O or be empty.

• **State:**

- ✓ Playerto movenext:X orO.
- ✓ Board configuration:



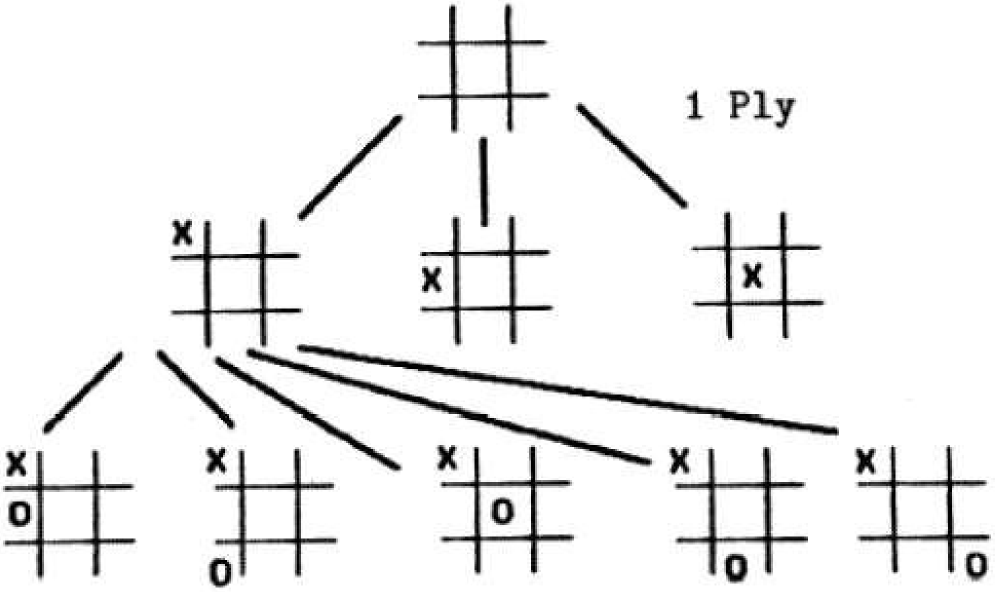
X		0
	0	
X		X

• **Operators:**Changeanemptycell to X orO.
• **StartState:**Boardempty;X’sturn.
• **TerminalStates:**ThreeX’sinarow; ThreeO’sina row;Allcellsfull.

SearchTree

Thesequenceofstatesformedbypossiblemovesiscalleda *searchtree*.Eachlevelofthetreeis called a *ply*.

Sincethesamestatemaybereachablebydifferent sequencesofmoves,the statespacemayin general be a graph. It may be treated as a tree for simplicity, at the cost of duplicating states.



Solvingproblemsusingsearch

- Givenan informaldescription oftheproblem,construct aformal descriptionasastatespace:
    - ✓ Defineadata structuretorepresentthe*state*.
    - ✓ Makearepresentationforthe *initialstate*fromthegiven data.
    - ✓ Writeprogramstorepresent*operators*thatchangeagivenstaterepresentationtoanew state representation.
    - ✓ Writeaprogram todetect*terminal states*.

- Chooseanappropriatesearchtechnique:
    - ✓ Howlargeis thesearch space?
    - ✓ Howwellstructured isthedomain?
    - ✓ Whatknowledge aboutthedomaincanbeused toguidethesearch?

HEURISTICSEARCHTECHNIQUES:

SearchAlgorithms

ManytraditionalsearchalgorithmsareusedinAIapplications.Forcomplex problems,the traditionalalgorithmsareunabletofindthesolutionswithinsomepracticaltimeandspace limits. Consequently, many special techniques are developed, using ***heuristic functions.*** Thealgorithms thatuse *heuristicfunctions*arecalled***heuristicalgorithms***.

• Heuristicalgorithmsarenotreallyintelligent;theyappeartobeintelligent becausethey achieve better performance.
• Heuristicalgorithmsaremoreefficientbecausetheytakeadvantageoffeedbackfromthedata to direct the search path.
• ***Uninformedsearchalgorithms***or*Brute-forcealgorithms*,searchthroughthesearchspaceall possible candidates for the solution checking whether each candidate satisfies the problem’s statement.
• ***Informed search algorithms*** use heuristic functions that are specific to the problem, apply themtoguidethesearch throughthesearchspacetotryto reducetheamountoftimespentin searching.

Agoodheuristicwillmakeaninformedsearchdramaticallyoutperformanyuninformedsearch: forexample,theTravelingSalesmanProblem(TSP),wherethe goalistofindisagoodsolution instead of finding the best solution.

In such problems, the search proceeds using current information about the problem to predict whichpathisclosertothegoalandfollowit,althoughitdoes notalwaysguaranteetofindthe best possible solution. Such techniques help in finding a solution within reasonable time and space (memory). Some prominent intelligent search algorithms are stated below:

1. ***GenerateandTestSearch***
2. ***Best-firstSearch***
3. ***GreedySearch***
4. ***A* Search***
5. ***Constraint Search***
6. ***Means-endsanalysis***

Therearesomemorealgorithms.Theyare either improvementsor combinations ofthese.
• **HierarchicalRepresentationofSearchAlgorithms:**AHierarchicalrepresentationofmost search algorithms is illustrated below. The representation begins with two types of search:
• **Uninformed Search:** Also called blind, exhaustive or brute-force search, it uses no informationabout the problem to guide thesearch and thereforemaynotbeveryefficient.
• **InformedSearch:**Alsocalledheuristicorintelligentsearch,thisusesinformationaboutthe problem to guide the search—usually guesses the distance to a goal state and is therefore efficient, but the search may not be always possible.

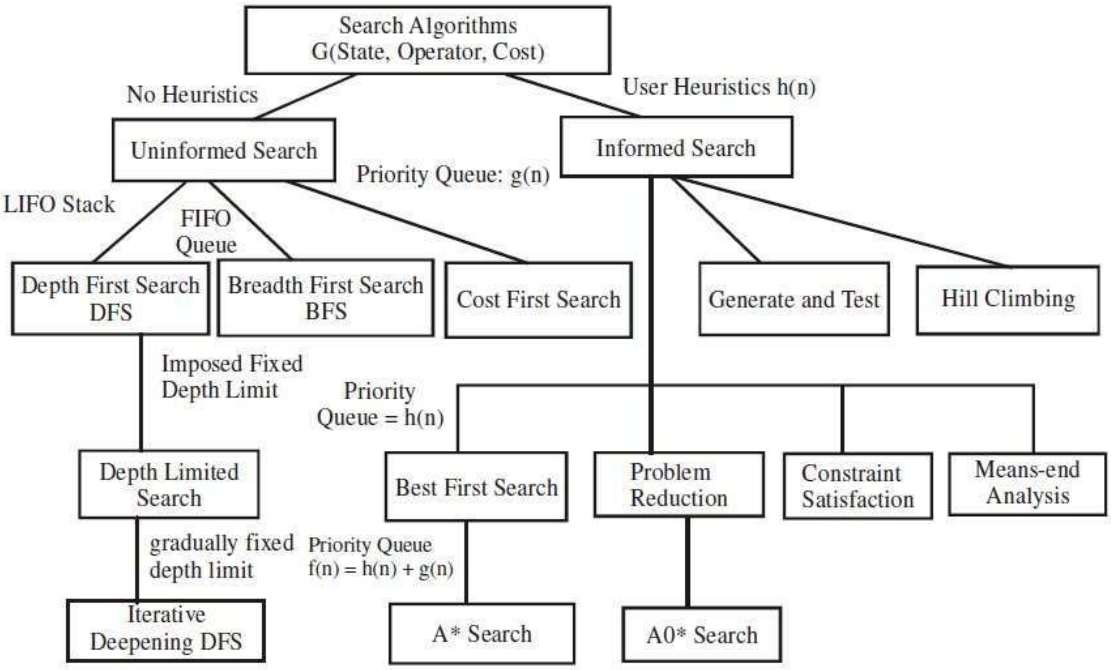
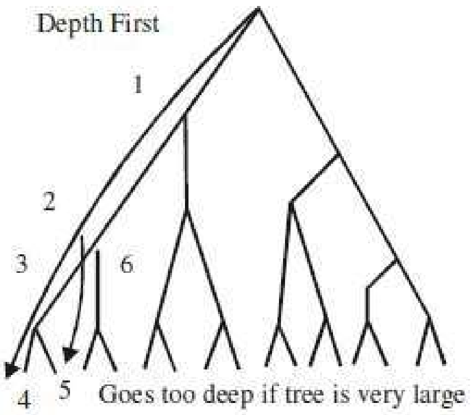
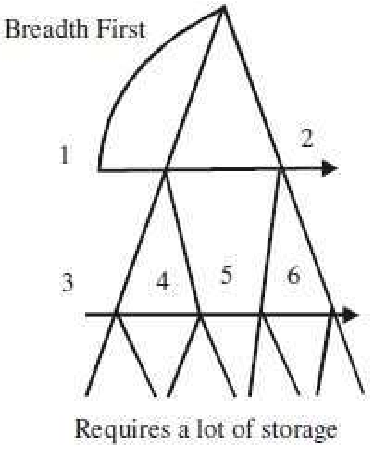


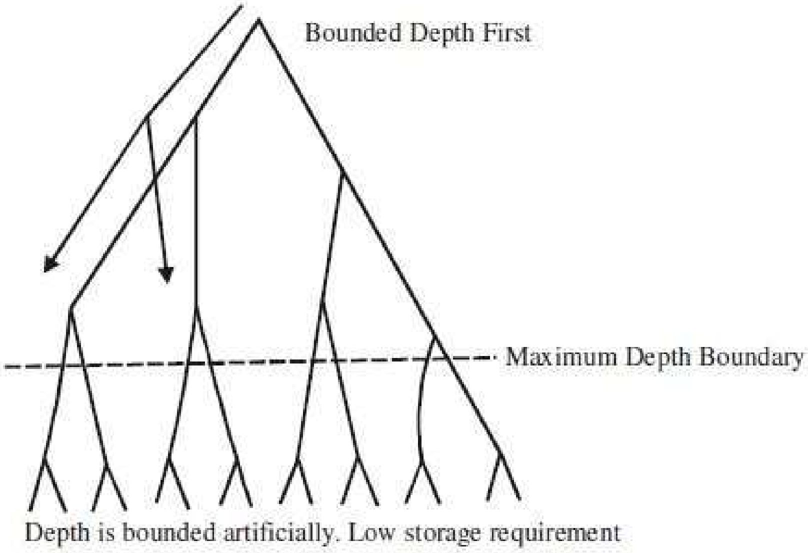
Fig. *Different Search Algorithms*

Thefirstrequirementisthatitcausesmotion,inagameplayingprogram,itmoves ontheboard and in the water jug problem, filling water is used to fill jugs. It means the control strategies without the motion will never lead to the solution.

The second requirement is that it is systematic, that is, it corresponds to the need for global motion as well as for local motion. This is a clear condition that neither would it be rational to fill a jug and empty it repeatedly, nor it would be worthwhile to move a piece round and round ontheboardinacyclicwayina game.Weshall initiallyconsidertwosystematicapproachesfor searching. Searches can be classified by the order in which operators are tried: depth-first, breadth-first, bounded depth-first.







Breadth-firstsearch

ASearchstrategy,inwhichthehighestlayerofadecisiontreeissearchedcompletelybefore proceeding to the next layer is called *Breadth-first search (BFS).*

• Inthisstrategy,noviablesolutionsareomittedandthereforeitisguaranteed thatanoptimal solution is found.

• Thisstrategyisoftennotfeasiblewhenthesearch spaceislarge.

Algorithm

1. Createavariable calledLISTandsetittobethestartingstate.
2. LoopuntilagoalstateisfoundorLISTis empty, Do
a. RemovethefirstelementfromtheLISTandcallitE.IftheLISTisempty,quit.
b. Foreverypatheachrulecanmatch thestateE,Do
(i) Applytheruletogenerate anewstate.
(ii) Ifthenew stateis a goalstate, quitand returnthis state.
(iii) Otherwise,add thenewstate to theend ofLIST.

Advantages

1. Guaranteedtofindanoptimalsolution(intermsofshortestnumberofsteps to reach the goal).
2. Canalwaysfind a goal nodeifoneexists (complete).

Disadvantages

1. Highstoragerequirement: *exponential*withtree depth.

Depth-firstsearch

Asearchstrategythatextendsthecurrentpathasfaraspossiblebeforebacktracking tothelast choice point and trying the next alternative path is called *Depth-first search (DFS).*

- Thisstrategydoesnotguaranteethattheoptimal solutionhasbeen found.
- Inthisstrategy,searchreachesasatisfactorysolutionmorerapidlythanbreadth first,an advantage when the search space is large.

Algorithm

Depth-firstsearchappliesoperatorstoeachnewlygeneratedstate,tryingtodrive directlytoward the goal.

1. Ifthestartingstateisagoalstate,quitand return success.
2. Otherwise,do the followinguntil success orfailureis signalled:

a. Generate asuccessor E to thestartingstate.Iftherearen o moresuccessors,thensignal failure.
b. CallDepth-firstSearchwithE asthestartingstate.
c. Ifsuccessisreturnedsignalsuccess;otherwise,continueinthe loop.

Advantages

1. Lowstoragerequirement:*linear*withtree depth.
2. Easilyprogrammed:functioncallstackdoesmostoftheworkofmaintainingstateofthe search.

Disadvantages

1. Mayfind asub-optimal solution (onethat is deeper ormore costlythan thebest solution).
2. Incomplete:without a depth bound,maynotfind asolution evenif one exists.

2.4.2.3Boundeddepth-firstsearch

Depth-first search can spend much time (perhaps infinite time) exploring a very deep path that doesnotcontainasolution,whenashallowsolutionexists.Aneasywayto solvethisproblemis to put a maximum depth bound on the search. Beyond the depth bound, a failure is generated automatically without exploring any deeper.

Problems:

1. It'shardtoguesshow deepthesolution lies.
2. Iftheestimateddepthistoodeep(evenby1)thecomputertimeusedis dramatically increased, by a factor of *bextra*.
3. Iftheestimateddepthistooshallow,thesearchfailstofindasolution;all thatcomputertime is wasted.

Heuristics

A heuristic is a method that improves the efficiency of the search process. These are like tour guides. There are good to the level that they may neglect the points in general interesting directions; they are bad to the level that they may neglect points of interest to particular individuals.Someheuristicshelpinthesearchprocesswithout sacrificinganyclaimstoentirety that the process might previously had. Others may occasionally cause an excellent path to be overlooked. By sacrificing entirety it increases efficiency. Heuristics may not find the best solutioneverytimebut guaranteethattheyfindagoodsolutioninareasonabletime.Theseare particularly useful in solving tough and complex problems, solutions of which would require infinite time, i.e. far longer than a lifetime for the problems which are not solved in any other way.

Heuristicsearch

To find a solution in proper time rather than a complete solution in unlimited time we use heuristics. 'A heuristic function is a function that maps from problem state descriptions to measures of desirability, usually represented as numbers'. Heuristic search methods use knowledge about the problem domain and choose promising operators first. These heuristic search methods use heuristic functions to evaluate the next state towards the goal state. For findingasolution,byusingtheheuristictechnique,oneshouldcarryoutthefollowingsteps:

1. Adddomain—specific informationto selectwhatis thebestpath tocontinuesearchingalong.
2. Defineaheuristicfunction h(n)that estimatesthe'goodness'ofanoden.
Specifically,h(n)=estimatedcost(ordistance)of minimalcostpathfrom n toagoal state.
3. The term, heuristic means 'serving to aid discovery' and is an estimate, based on domain specificinformationthatiscomputablefromthecurrentstatedescriptionofhowcloseweareto a goal.

Findingaroutefromonecitytoanothercityisanexampleofasearchproblemin which different search orders and the use of heuristic knowledge are easily understood.

1. State:Thecurrent cityin which thetraveller is located.
2. Operators:Roadslinkingthe current citytoother cities.
3. CostMetric:Thecostof takinga givenroadbetween cities.
4. Heuristicinformation: Thesearchcouldbeguidedbythedirectionofthe goal cityfromthe current city, or we could use airline distance as an estimate of the distance to the goal.

Heuristic search techniques

For complex problems, the traditional algorithms, presented above, are unable to find the solutionwithinsomepracticaltimeandspacelimits.Consequently,manyspecialtechniquesare developed, using ***heuristic functions.***

- Blindsearchisnotalwayspossible,becauseitrequirestoomuchtimeorSpace (memory).

Heuristicsare***rules ofthumb***;theydo notguaranteeasolution to a problem.

- HeuristicSearchisaweaktechniquebutcanbeeffectiveifappliedcorrectly; itrequires domain specific information.

Characteristicsofheuristicsearch

- Heuristicsareknowledge aboutdomain,whichhelp searchandreasoninginits domain.
- Heuristicsearchincorporatesdomainknowledgeto improveefficiencyover blind search.
- Heuristicisafunctionthat,whenappliedtoastate,returnsvalueasestimatedmeritofstate, with respect to goal.
    - ✓ Heuristicsmight(forreasons)*underestimate*or*overestimate*themeritofastatewith respect to goal.
    - ✓ Heuristicsthatunderestimatearedesirable andcalled admissible.
- Heuristicevaluationfunctionestimateslikelihoodofgivenstateleadingtogoalstate.
- Heuristicsearchfunctionestimatescostfromcurrentstatetogoal,presumingfunctionis efficient.

Heuristicsearchcomparedwithother search
TheHeuristicsearchiscomparedwithBruteforceorBlindsearchtechniques below:

ComparisonofAlgorithms

Bruteforce/Blind search	**Heuristicsearch**
Can only search what it has knowledge about already	Estimates'distance'togoalstate through explored nodes
No knowledge about how far a node node from goal state	Guidessearchprocesstowardgoal
	Prefers states (nodes) that lead closetoandnotawayfromgoal state

Example:Travellingsalesman
A salesman has to visit a list of cities and he must visit each cityonlyonce. There are different routesbetweenthecities.Theproblemistofindtheshortestroutebetweenthecitiessothatthe salesman visits all the cities at once.

SupposethereareNcities,thenasolutionwouldbetotakeN!possible combinationstofindthe shortest distance to decide the required route. This is not efficient as with N=10 there are 36,28,800 possible routes. This is an example of *combinatorial explosion*.

There are better methods for the solution of such problems: one is called *branch* and *bound*. First,generateallthecompletepathsandfindthedistanceofthefirstcompletepath.Ifthenext path is shorter, then save it and proceed this wayavoidingthe path when its length exceeds the saved shortest path length, although it is better than the previous method.

GenerateandTestStrategy

Generate-And-TestAlgorithm
Generate-and-testsearchalgorithmisaverysimplealgorithmthatguaranteestofindasolutionif done systematically and there exists a solution.

Algorithm:Generate-And-Test

1. Generateapossiblesolution.
2. Testto seeif this istheexpected solution.
3. Ifthesolutionhasbeenfoundquitelsegotostep1.

Potentialsolutionsthatneedtobegeneratedvarydependingonthekindsofproblems.Forsome problems the possible solutions may be particular points in the problem space and for some problems, paths from the start state.

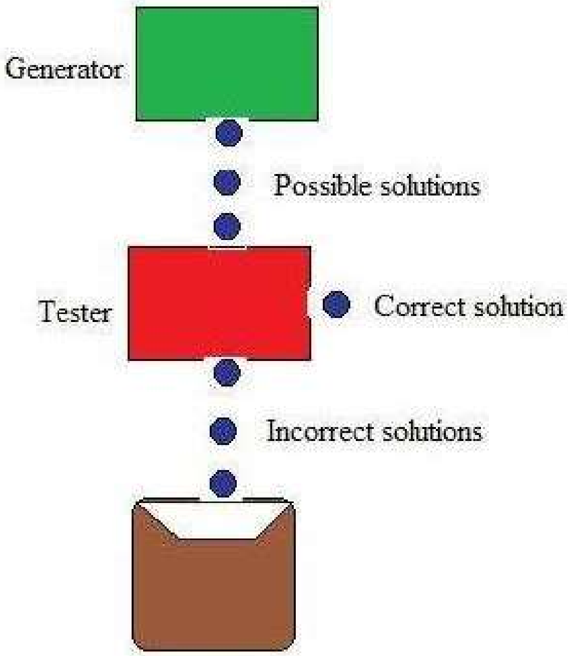


Figure:GenerateAndTest

Generate-and-test,likedepth-firstsearch,requiresthatcompletesolutionsbegeneratedfor testing. In its most systematic form, it is only an exhaustive search of the problem space. Solutionscanalsobegeneratedrandomlybutsolutionisnotguaranteed.Thisapproachiswhatis known as British Museum algorithm: finding an object in the British Museum by wandering randomly.

SystematicGenerate-And-Test

Whilegeneratingcompletesolutionsandgeneratingrandomsolutionsarethetwoextremesthere exists another approach that lies in between. The approach is that the search process proceeds systematically but some paths that unlikely to lead the solution are not considered. This evaluation is performed by a heuristic function.

Depth-firstsearchtreewithbacktrackingcanbeusedtoimplementsystematic generate-and-test procedure. As per this procedure, if some intermediate states are likely to appear often in the tree, it would be better to modify that procedure to traverse a graph rather than a tree.

Generate-And-TestAnd Planning

Exhaustivegenerate-and-testisveryusefulforsimpleproblems.Butforcomplexproblemseven heuristic generate-and-test is not very effective technique. But this may be made effective by combiningwithothertechniquesinsuchawaythatthespaceinwhichtosearchisrestricted.An AIprogramDENDRAL, forexample,usesplan-Generate-and-testtechnique.First,theplanning process uses constraint-satisfaction techniques and creates lists of recommended and contraindicated substructures. Then the generate-and-test procedure uses the lists generated and required to explore only a limited set of structures. Constrained in this way, generate-and-test proved highly effective. A major weakness of planning is that it often produces inaccurate solutions as there is no feedback from the world. But if it is used to produce only pieces of solutions then lack of detailed accuracy becomes unimportant.

HillClimbing

HillClimbingisheuristicsearchusedformathematicaloptimizationproblemsinthefieldof Artificial Intelligence .

Givenalargesetofinputsandagoodheuristicfunction,ittriestofindasufficientlygood solution to the problem. This solution may not be the global optimal maximum.

- In the above definition, mathematical optimization problems implies that hill climbing solves the problems where we need to maximize or minimize a given real function by choosingvaluesfromthegiveninputs.Example-Travellingsalesmanproblemwherewe need to minimize the distance traveled by salesman.
- 'Heuristicsearch'meansthatthissearchalgorithmmaynotfindtheoptimalsolutionto the problem. However, it will give a good solution in reasonable time.
- A heuristic function is a function that will rank all the possible alternatives at any branchingstepinsearchalgorithmbasedontheavailableinformation.Ithelpsthe algorithm to select the best route out of possible routes.

FeaturesofHillClimbing

1. Variantofgenerateandtestalgorithm: Itisavariantofgenerateandtestalgorithm.The generate and test algorithm is as follows :

1. Generateapossiblesolutions.
2. Testto seeif thisistheexpected solution.
3. Ifthe solutionhas beenfound quitelsegoto step 1.

Hence we call Hill climbing as a variant of generate and test algorithm as it takes the feedback fromtestprocedure.Thenthisfeedbackisutilizedbythegeneratorindecidingthenextmovein search space.

2. UsestheGreedyapproach:Atanypointinstatespace,thesearchmovesin thatdirection onlywhichoptimizesthecostoffunctionwiththe hopeoffindingtheoptimalsolutionat the end.

TypesofHillClimbing

1. SimpleHillclimbing: Itexaminestheneighboringnodesonebyoneandselectsthefirst neighboring node which optimizes the current cost as next node.
   AlgorithmforSimpleHillclimbing :

Step1:Evaluatetheinitialstate.Ifitisagoalstatethenstopandreturnsuccess.Otherwise, make initial state as current state.
Step2:Loopuntilthesolutionstateis foundortherearenonewoperatorspresentwhichcanbe applied to current state.
a) Selectastatethathasnotbeenyetappliedtothecurrentstateandapplyittoproduceanew state.
b) Performthesetoevaluatenew state
i. Ifthecurrent state isa goal state, thenstop and return success.
ii. Ifitis betterthanthecurrentstate,then makeitcurrent stateandproceed further.
iii. Ifit is notbetter than thecurrent state, thencontinuein theloop until asolution is found.

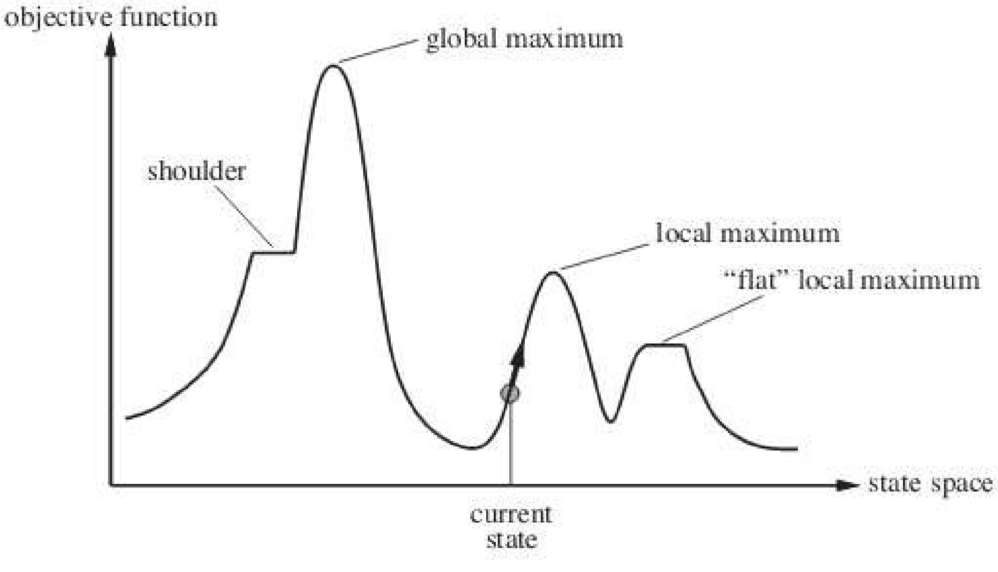
Step 3:Exit.

2. Steepest-AscentHillclimbing: Itfirstexaminesalltheneighboringnodesandthen selects the node closest to the solution state as next node.

Step1:Evaluatetheinitialstate.Ifitisgoalstatethenexitelsemakethecurrentstateasinitial state
Step2 :Repeatthesesteps untila solutionisfound or currentstate does not change
i. Let'target'bea statesuchthat anysuccessorofthe currentstatewill bebetterthan it;
ii. foreachoperator thatappliesto thecurrent state
a. applythenewoperator andcreate anew state
b. evaluatethenewstate
c. ifthisstateisgoal statethen quit elsecomparewith 'target'
d. ifthisstateisbetterthan'target',set thisstateas'target'
e. iftargetisbetterthancurrentstatesetcurrentstatetoTarget Step 3
: Exit

3. Stochastic hill climbing : It does not examine all the neighboring nodes before deciding whichnodetoselect.Itjustselectsaneighboringnodeatrandom,anddecides(basedon the amount of improvement in that neighbor) whether to move to that neighbor or to examine another.

StateSpacediagramfor Hill Climbing
Statespacediagramisagraphicalrepresentationofthesetofstatesoursearchalgorithmcan reach vs the value of our objective function(the function which we wish to maximize).
X-axis:denotesthestatespaceiestates orconfiguration ouralgorithm mayreach.
Y-axis:denotesthevalues ofobjectivefunctioncorrespondingto toaparticular state.
Thebestsolutionwillbethatstatespacewhereobjectivefunctionhasmaximumvalue(global maximum).



DifferentregionsintheStateSpace Diagram

1. Local maximum : It is a state which is better than its neighboring state however there existsastatewhichisbetterthanit(globalmaximum).Thisstateisbetterbecausehere value of objective function is higher than its neighbors.

2. Globalmaximum:Itisthebestpossiblestateinthestatespacediagram.Thisbecauseat this state, objective function has highest value.
3. Plateua/flatlocalmaximum: Itisaflatregionofstatespacewhereneighboringstates have the same value.
4. Ridge: Itisregionwhichishigherthanitsneighboursbutitselfhasaslope.Itisaspecial kind of local maximum.
5. Currentstate:Theregionofstatespacediagramwherewearecurrentlypresentduring the search.
6. Shoulder:Itisaplateauthathasanuphilledge.

Problems in different regions in Hill climbing
Hillclimbingcannotreachtheoptimal/beststate(globalmaximum)ifitentersanyofthe following regions :

1. Local maximum : At a local maximum all neighboring states have a values which is worse than than the current state. Since hill climbinguses greedyapproach, it will not movetotheworsestateandterminateitself.Theprocesswillendeventhoughabetter solution may exist.
   Toovercomelocalmaximumproblem:Utilizebacktrackingtechnique.Maintainalistof visited states. If the search reaches an undesirable state, it can backtrack to the previous configuration and explore a new path.
2. Plateau:Onplateauallneighborshavesamevalue.Hence,itisnotpossibletoselectthe best direction.

Toovercomeplateaus: Makeabigjump.Randomlyselect astatefarawayfromcurrentstate. Chances are that we will land at a non-plateau region

3. Ridge:Anypointonaridgecanlooklikepeakbecausemovementinallpossible directions is downward. Hence the algorithm stops when it reaches this state.
   ToovercomeRidge: Inthiskindofobstacle,usetwoormorerulesbeforetesting.It implies moving in several directions at once.

BestFirstSearch(InformedSearch)

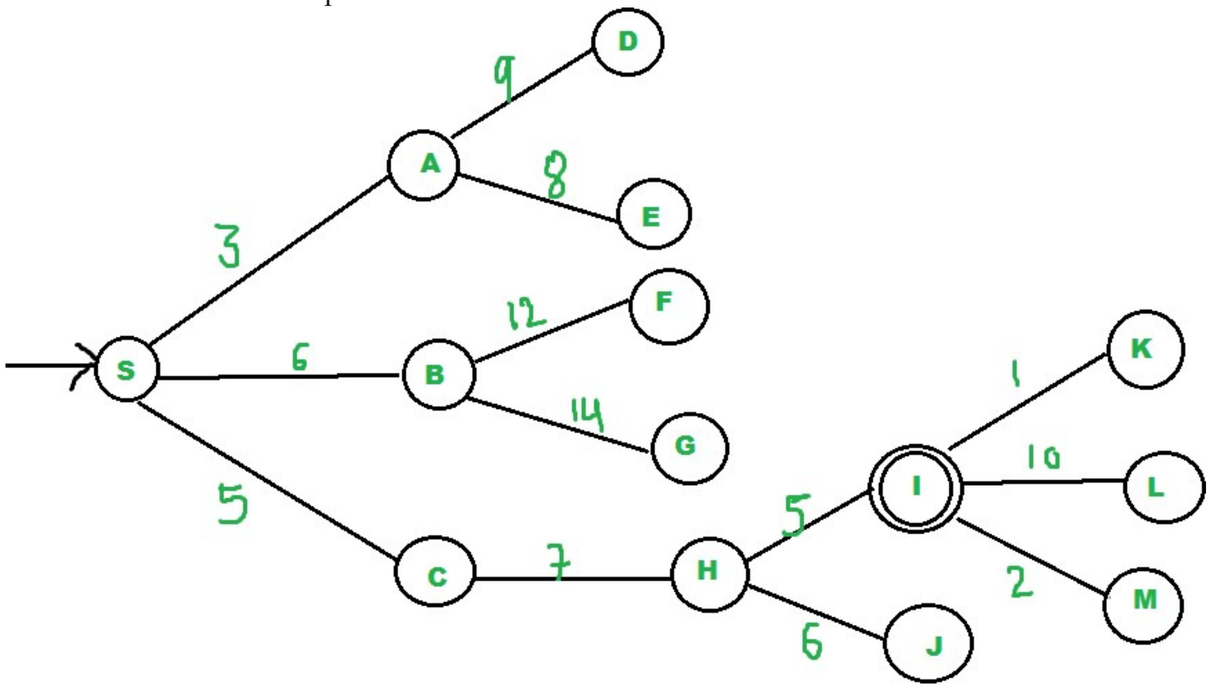
In BFS and DFS, when we are at a node, we can consider any of the adjacent as next node.SobothBFSandDFSblindlyexplorepathswithoutconsideringanycostfunction.The idea of Best First Search is to use an evaluation function to decide which adjacent is most promising and then explore. Best First Search falls under the categoryof Heuristic Search or Informed Search.
Weuseapriorityqueuetostorecostsofnodes.So theimplementationisavariationofBFS,we just need to change Queue to PriorityQueue.

Algorithm:

```
Best-First-Search(Grahg,Nodestart)
   1) CreateanemptyPriorityQueue
      PriorityQueue pq;
   2) Insert"start"inpq.
      pq.insert(start)
   3) Until PriorityQueue is empty
         u=PriorityQueue.DeleteMin
```

```
        Ifuisthegoal Exit
        Else
          Foreachneighborvofu If
            v "Unvisited"
              Markv"Visited"pq.i
              nsert(v)
          Markv"Examined"
End procedure
```

Letus considerbelow example.



Westartfromsource"S"andsearchfor goal "I" using given costs and BestFirst search.

pq initiallycontainsS
Weremovesfromandprocessunvisited neighbors of S to pq.
pqnowcontains{A,C,B}(Cisput before B because C has lesser cost)

WeremoveAfrompqandprocessunvisited neighbors of A to pq.
pqnow contains{C, B,E, D}

WeremoveCfrompqandprocessunvisited
neighbors of C to pq.
pqnowcontains {B,H, E, D}

WeremoveBfrompqandprocessunvisited
neighbors of B to pq.
pqnowcontains {H,E,D,F, G}

WeremoveHfrompq.Sinceourgoal "I" is
a neighbor of H, we return.

Analysis :

- TheworstcasetimecomplexityforBest FirstSearchisO(n*Logn)wherenisnumber of nodes. In worst case, we may have to visit all nodes before we reach goal. Note that priority queue is implemented using Min(or Max) Heap, and insert and remove operations take O(log n) time.
- Performanceofthealgorithmdependsonhowwellthecostorevaluationfunctionis designed.

A*SearchAlgorithm

A* is a type of search algorithm. Some problems can be solved by representing the world in the initialstate,andthenforeachactionwecanperformontheworldwegeneratestatesforwhatthe world would be like if we did so. If you do this until the world is in the state that we specified as a solution, then the route from the start to this goal state is the solution to your problem.

InthistutorialIwilllookattheuseofstatespacesearchtofindtheshortestpathbetweentwo points (pathfinding), and also to solveasimple slidingtile puzzle (the 8-puzzle). Let's look at some of the terms used in Artificial Intelligence when describing this state space search.

Some terminology

A *node* is a state that the problem's world can be in. In pathfinding a node would be just a 2d coordinateofwhereweareatthepresenttime. In the8-puzzleitisthepositionsofallthetiles. Next all the nodes are arranged in a *graph* where links between nodes represent valid steps in solving the problem. These links are known as *edges*. In the 8-puzzle diagram the edges are shown as blue lines. See figure 1 below.
Statespacesearch,then,issolvingaproblembybeginningwiththestartstate,andthenforeach node weexpand all the nodes beneath it in the graph byapplyingall the possible moves that can be made at each point.

HeuristicsandAlgorithms

Atthispointweintroduceanimportantconcept,the*heuristic*.Thisislikeanalgorithm,butwith a keydifference. An algorithm is a set of steps which you can follow to solve a problem, which always works for valid input. For example you could probably write an algorithm yourself for

multiplyingtwonumberstogetheronpaper.Aheuristicisnotguaranteedtoworkbutisusefulin that it may solve a problem for which there is no algorithm.
Weneedaheuristictohelpuscutdownonthishugesearchproblem.What weneedistouseour heuristic at each node to make an estimate of how far we are from the goal. In pathfinding we know exactly how far we are, because we know how far we can move each step, and we can calculate the exact distance to the goal.
But the 8-puzzle is more difficult. There is no known algorithm for calculating from a given position how many moves it will take to get to the goal state. So various heuristics have been devised.Thebestonethat IknowofisknownastheNilssonscorewhichleadsfairlydirectlyto the goal most of the time, as we shall see.

Cost

When lookingat each node in the graph, we now have an idea of a heuristic, which can estimate how close the state is to the goal. Anotherimportant consideration is the cost of gettingto where we are. In the case of pathfinding we often assign a movement cost to each square. The cost is the same then the cost of each square is one. If we wanted to differentiate between terrain types wemaygivehighercoststograss andmudthantonewlymaderoad.Whenlookingatanodewe want to add up the cost of what it took to get here, and this is simplythe sum of the cost of this node and all those that are above it in the graph.

8 Puzzle

Let's look at the8 puzzle in moredetail. This is asimple slidingtile puzzle on a3*3 grid where onetileismissingandyoucanmovetheothertilesintothegapuntilyougetthepuzzleintothe goal position. See figure 1.

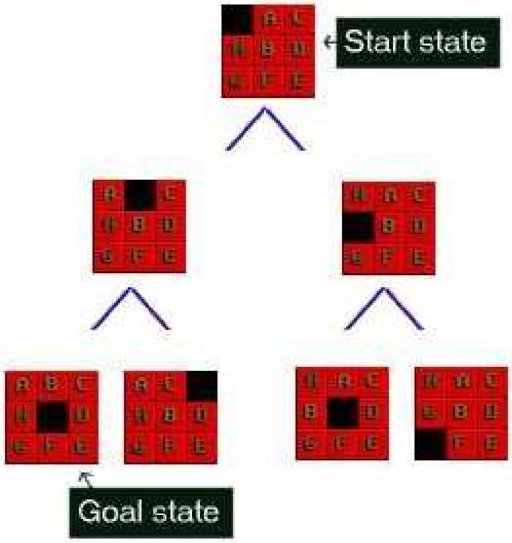


Figure 1 :The8-Puzzlestatespacefora verysimple example

Thereare362,880differentstatesthatthepuzzlecanbein,andtofindasolutionthesearchhas to find a route through them. From most positions of the search the number of edges (that's the bluelines)istwo.Thatmeansthatthenumberofnodesyouhaveineachleveloftthesearchis 2^d where d is the depth. If the number of steps to solve a particular state is 18, then that�s 262,144 nodes just at that level.

The 8 puzzle game state is as simple as representing a list of the 9 squares and what's in them. Herearetwostatesforexample;thelastoneistheGOALstate, atwhichpointwe'vefoundthe solution. The first is a jumbled up example that you may start from.

StartstateSPACE,A,C,H,B,D,G,F,E Goal
state A, B, C, H, SPACE, D, G, F, E
Therulesthat youcanapplytothepuzzlearealsosimple.Ifthereisablanktileabove,below,to the left or to the right of a given tile, then you can move that tile into the space. To solve the puzzle you need to find the path from the start state, through the graph down to the goal state.

Thereis examplecodetoto solve the8-puzzleonthe githubsite.

Pathfinding

In a video game, or some other pathfinding scenario, you want to search a state space and find out how to get from somewhere you are to somewhere you want to be, without bumping into wallsorgoingtoofar.Forreasonswewillseelater,theA*algorithmwillnotonlyfindapath,if there is one, but it will find the shortest path. A state in pathfinding is simply a position in the world. In the example of a maze game like Pacman you can represent where everything is using asimple 2d grid. Thestart state for a ghost say, would be the2d coordinate of wherethe ghost is at the start of the search. The goal state would be where pacman is so we can go and eat him.
Thereis also examplecodetodo pathfindingon thegithubsite.

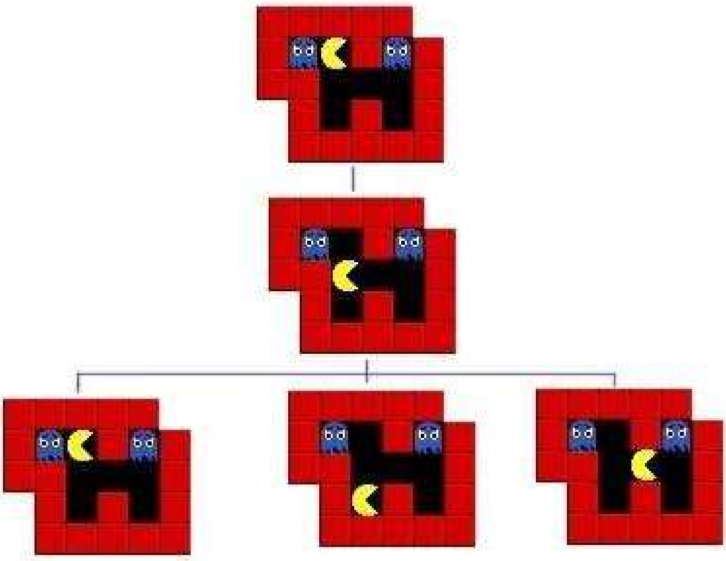


Figure2 :Thefirst threesteps ofa pathfinding state space

ImplementingA*

WearenowreadytolookattheoperationoftheA*algorithm.Whatweneedtodoisstartwith the goal state and then generate the graph downwards from there. Let's take the 8-puzzle in figure 1. We ask how manymoves can we make from the start state? The answer is 2, there are two directions we can move the blank tile, and so our graph expands.

If we were just to continue blindly generating successors to each node, we could potentially fill thecomputer'smemorybeforewefoundthegoalnode.Obviouslyweneedtorememberthebest nodes and search those first. We also need to remember the nodes that we have expanded already, so that we don't expand the same state repeatedly.

Let's start with the OPEN list. This is where we will remember which nodes we haven't yet expanded. When the algorithm begins thestart stateis placed on the open list, it is the onlystate weknowaboutandwehavenotexpandedit.Sowewillexpandthenodesfromthestartandput those on the OPEN list too. Now we are done with the start node and we will put that on the CLOSED list. The CLOSED list is a list of nodes that we have expanded.

$f = g + h$

Using the OPEN and CLOSED list lets us be more selective about what we look at next in the search. Wewant to look at the best nodes first. Wewill give each nodeascoreon how good we think it is. This score should be thought of as the cost of getting from the node to the goal plus thecostof gettingtowhereweare.Traditionallythishas beenrepresentedbythelettersf, gand h. 'g' is the sum of all the costs it took to get here, 'h' is our heuristic function, the estimate of whatitwilltaketogetto thegoal.'f'isthesumofthesetwo.Wewillstoreeachoftheseinour nodes. Usingthef,gandhvaluestheA*algorithmwillbedirected,subjecttoconditionswewilllook at further on, towards the goal and will find it in the shortest route possible.

SofarwehavelookedatthecomponentsoftheA*,let'sseehowtheyallfit togethertomakethe algorithm :

Pseudocode

Hopefullytheideaswelookedatintheprecedingparagraphswillnowclickintoplaceaswe look at the A* algorithm pseudocode. You may find it helpful to print this out or leave the window open while we discuss it.

Tohelpmaketheoperationofthealgorithmclearwewilllookagainatthe8-puzzleproblemin figure 1 above. Figure 3 below shows the f,g and h scores for each of the tiles.

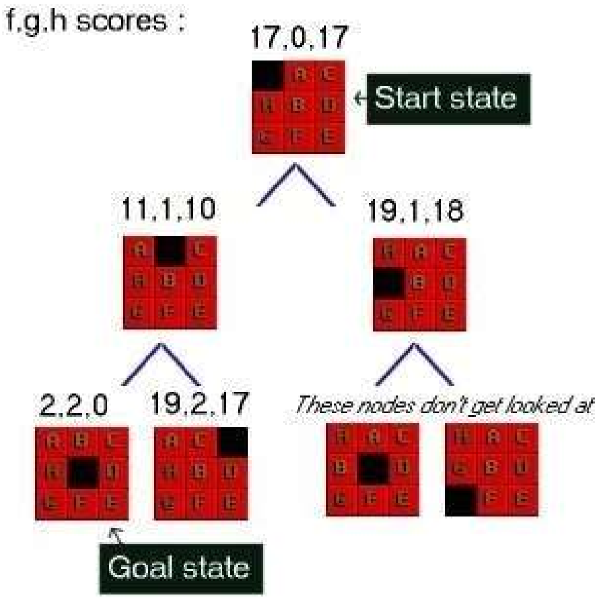


Figure3 :8-Puzzlestate spaceshowing f,g,h scores

Firstofalllookatthegscoreforeachnode.Thisisthecostofwhatittooktogetfromthestart tothatnode.Sointhepicturethecenternumberisg.As youcanseeitincreasesbyoneateach level. In some problems the cost may vary for different state changes. For example in pathfinding there is sometimes a type of terrain that costs more than other types.
Next look at the last number in each triple. This is h, the heuristic score. As Imentioned above I amusingaheuristicknownasNilsson'sSequence,whichconvergesquicklytoacorrectsolution in many cases. Here is how you calculate this score for a given 8-puzzle state :

Advantages:

Itiscompleteandoptimal.

Itisthebestone fromothertechniques.Itisused tosolveverycomplexproblems.

Itisoptimallyefficient,i.e.thereisnootheroptimalalgorithmguaranteedtoexpandfewernodes than A*.

Disadvantages:

Thisalgorithm iscomplete ifthebranchingfactorisfinite andeveryactionhas fixed cost.

ThespeedexecutionofA*searchishighlydependantontheaccuracyoftheheuristicalgorithm that is used to compute h (n).

AO*Search:(And-Or)Graph

TheDepthfirstsearchandBreadthfirstsearchgivenearlierforORtreesorgraphscanbeeasily adopted by AND-OR graph. The main difference lies in the way termination conditions are determined, since all goals following an AND nodes must be realized; where as a single goal node following an OR node will do. So for this purpose we are using AO* algorithm.

LikeA*algorithmherewewillusetwoarraysandoneheuristicfunction.

OPEN:

Itcontainsthenodes thathasbeentraversed butyetnotbeen markedsolvableor unsolvable.

CLOSE:

Itcontains thenodes thathavealreadybeen processed.

67: Thedistancefromcurrentnodetogoalnode.

Algorithm:

Step1: Placethe startingnodeinto OPEN.

Step2: Computethe most promisingsolution treesay T0.

Step3: SelectanodenthatisbothonOPENandamemberofT0.RemoveitfromOPENand place it in

CLOSE

Step4: Ifnistheterminalgoalnodethenlevelednassolvedandleveledalltheancestorsofn as solved. If the starting node is marked as solved then success and exit.

Step5: Ifnisnotasolvablenode,thenmarknasunsolvable.Ifstartingnodeismarkedas unsolvable, then return failure and exit.

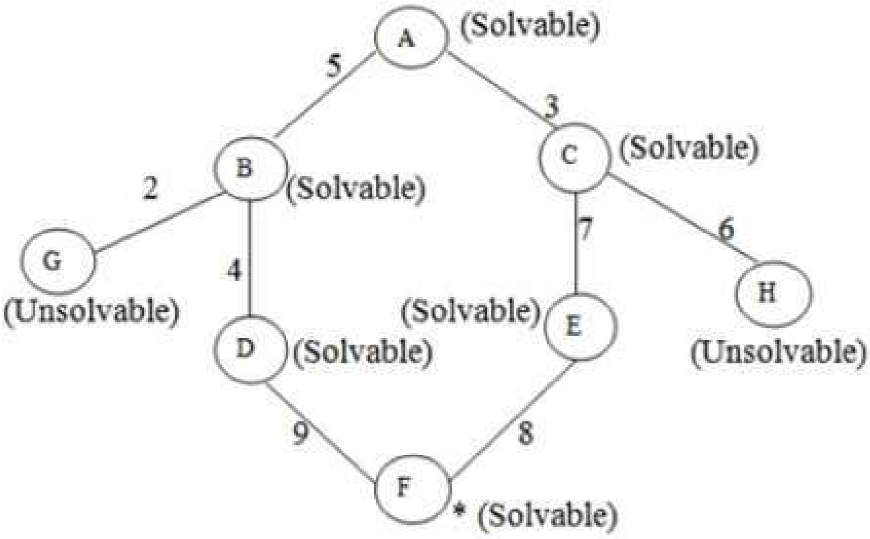
Step6: Expand n.Find all itssuccessors and findtheirh (n)value,push them into OPEN.

Step7: Return toStep 2.

Step8: Exit.

Implementation:

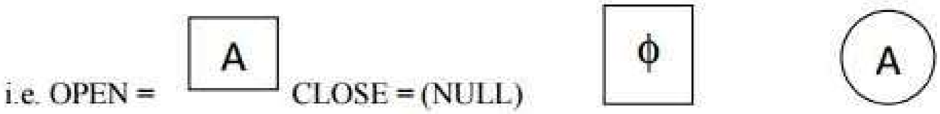
Letustakethe followingexampleto implementtheAO* algorithm.



Figure

Step1:

Intheabovegraph,thesolvablenodesareA,B,C,D,E,Fandtheunsolvablenodesare G,H. Take A as the starting node. So place A into OPEN.



i.e. OPEN = A CLOSE = (NULL) ϕ

Step 2:

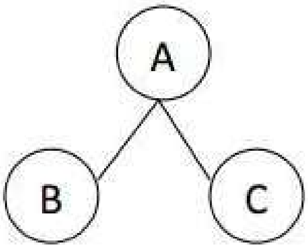
The children of A are B and C which are solvable. So place them into OPEN and place A into the CLOSE.

i.e. OPEN =

B	C

CLOSE =

A



Step 3:

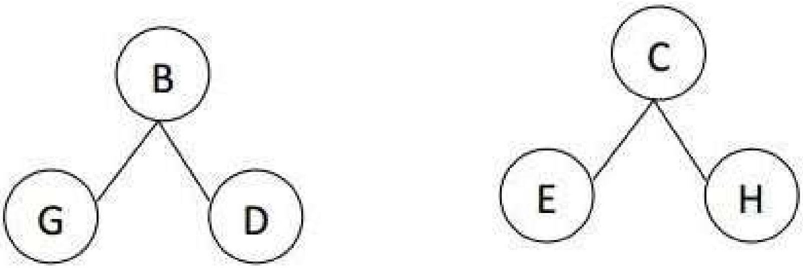
Now process the nodes B and C. The children of B and C are to be placed into OPEN. Also remove B and C from OPEN and place them into CLOSE.

So OPEN =

G	D	E	

C

A	B	C



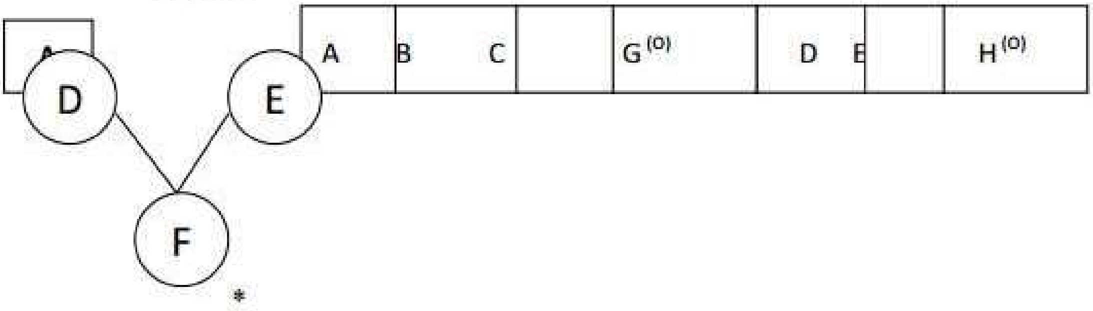
(O)

‘O’ indicated that the nodes G and H are unsolvable.

Step 4:

As the nodes G and H are unsolvable, so place them into CLOSE directly and process the nodes D and E.

i.e. OPEN = CLOSE =

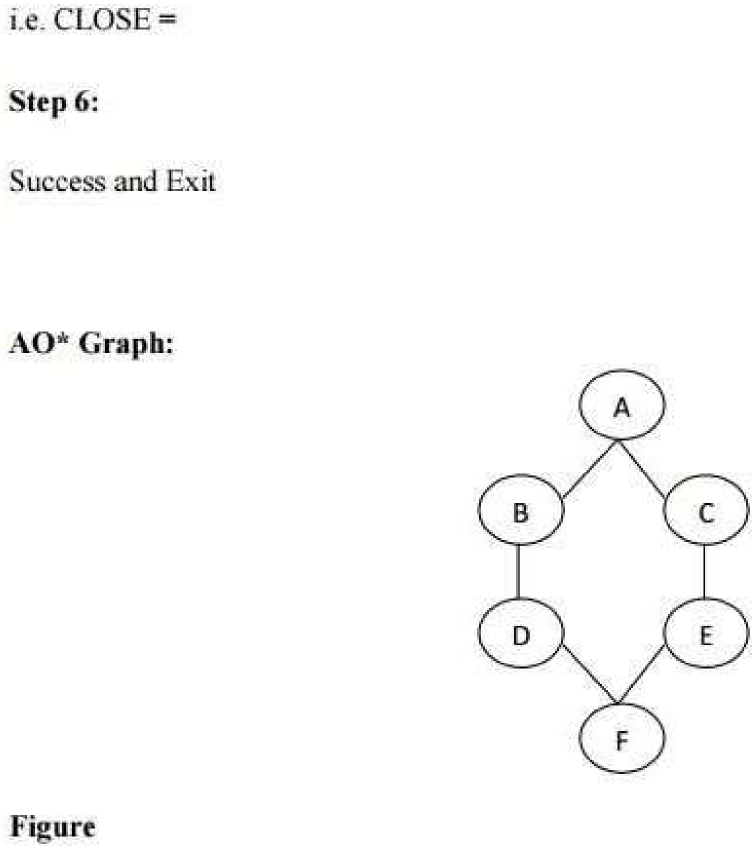


A	B C		$G^{(O)}$	D E		$H^{(O)}$

Step 5:

Now we have been reached at our goal state. So place F into CLOSE.

A	B C		$G^{(O)}$	D E		$H^{(O)}$	F



i.e. CLOSE =

Step 6:

Success and Exit

AO* Graph:

Figure

Advantages:

Itisanoptimalalgorithm.

Iftraverseaccordingto theorderingof nodes.Itcan beused forboth ORand ANDgraph.

Disadvantages:

Sometimesforunsolvablenodes,itcan'tfindtheoptimalpath.Itscomplexityisthanother algorithms.

PROBLEMREDUCTION

ProblemReductionwithAO*Algorithm.

When a problem can be divided into a set of sub problems, where each sub problem can be solvedseparatelyandacombinationofthesewillbeasolution,AND-OR graphsorAND-OR trees are used for representing the solution. The decomposition of the problem or problem reduction generates AND arcs. OneAND aremaypoint to anynumberof successor nodes. All thesemustbesolvedsothatthearcwillrisetomanyarcs,indicatingseveralpossiblesolutions. Hence the graph is known as AND - OR instead of AND. Figure shows an AND - OR graph.

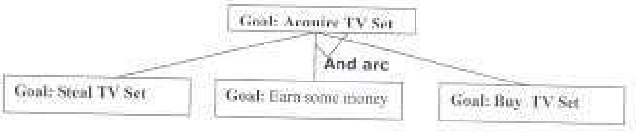


Figure shows AND - Or graph - an example.

AnalgorithmtofindasolutioninanAND-ORgraphmusthandleANDareaappropriately.A* algorithm can not search AND - OR graphs efficiently. This can be understand from the give figure. FIGURE:AND-ORgraph

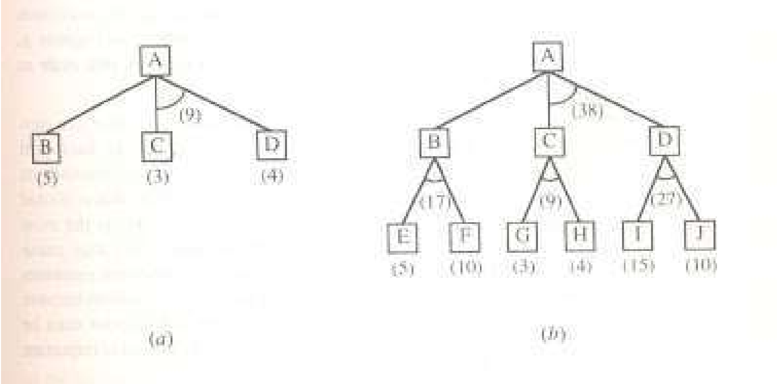


Figure 3.7: AND-OR Graphs

Infigure(a)thetopnodeAhasbeenexpandedproducingtwoareaoneleadingtoBandleading to C-D . the numbers at each node represent the value of f ' at that node (cost of getting to the goal state from current state). For simplicity, it is assumed that every operation(i.e. applying a rule) has unit cost, i.e., each are with single successor will have a cost of 1 and each of its components. With the available information till now , it appears that C is the most promising node to expand since its f ' = 3 , the lowest but going through B would be better since to use C we must also use D' and the cost would be 9(3+4+1+1). Through B it would be 6(5+1).

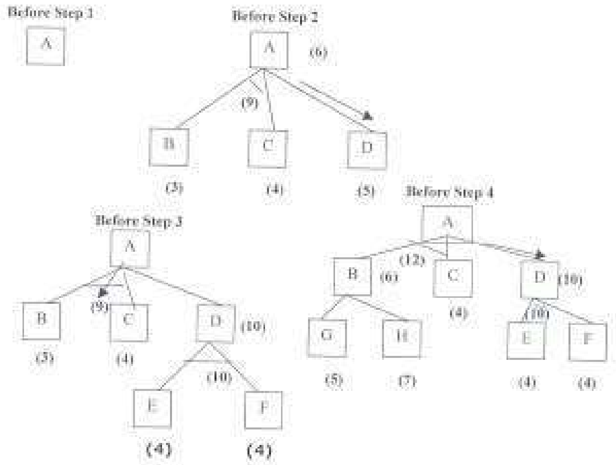
Thus the choice of the next node to expand depends not only n a value but also on whether that nodeispartofthecurrentbestpathformtheinitialmode.Figure(b)makesthisclearer. Infigure the node G appears to be the most promising node, with the least f ' value. But G is not on the current beat path, since to use G we must use GH with a cost of 9 and again this demands that arcs be used (with a cost of 27). The path from A through B, E-F is better with a total cost of (17+1=18). Thus we can see that to search an AND-OR graph, the following three things must be done.

1. traversethegraphstartingattheinitialnodeandfollowingthecurrentbestpath,and accumulate the set of nodes that are on the path and have not yet been expanded.

2. Pickoneoftheseunexpandednodesandexpandit.Additssuccessorstothegraphand computer f ' (cost of the remaining distance) for each of them.

3. Changethef'estimateofthenewlyexpandednodetoreflectthenewinformationproduced by its successors. Propagate this change backward through the graph. Decide which of the current best path.

The propagation of revised cost estimation backward is in the tree is not necessary in A* algorithm.ThisisbecauseinAO*algorithmexpandednodesarere-examinedsothatthecurrent best path can be selected. The working of AO* algorithm is illustrated in figure as follows:



Referringthefigure.TheinitialnodeisexpandedandDisMarkedinitiallyaspromisingnode. D is expanded producing an AND arcE-F. f 'value of Dis updated to 10. Goingbackwards wecan see that the AND arc B-C is better . it is now marked as current best path. B and C have to be expandednext.Thisprocesscontinuesuntilasolutionisfoundorallpathshaveledtodeadends, indicating that there is no solution. An A* algorithm the path from one node to the other is always that of the lowest cost and it is independent of the paths through other nodes.

The algorithm for performing a heuristic search of an AND - OR graph is given below. Unlike A* algorithm which used two lists OPEN and CLOSED, the AO* algorithm uses a single structure G. G represents the part of the search graph generated so far. Each node in G points down to its immediate successors and up to its immediate predecessors, and also has with it the value of h' cost of a path from itself to a set of solution nodes. The cost of getting from the start nodes to the current node "g" is not stored as in the A* algorithm. This is because it is not possibletocomputeasinglesuch valuesincetheremaybemanypaths tothesamestate.In AO* algorithm serves as the estimate of goodness of a node. Also a there should value called FUTILITY is used. The estimated cost of a solution is greaterthan FUTILITY then the search is abandoned as too expansive to be practical.
Forrepresentingabove graphsAO*algorithmisasfollows

AO*ALGORITHM:

1. LetGconsistsonlytothenoderepresentingtheinitialstatecallthisnodeINTT.Compute h' (INIT).

2. UntilINITislabeledSOLVEDorhi(INIT)becomesgreaterthanFUTILITY,repeatthe following procedure.

(I) TracethemarkedarcsfromINITandselectanunboundednode NODE.

(II) Generate the successors ofNODE . if there are no successors then assign FUTILITY as h'(NODE).ThismeansthatNODEisnotsolvable.Ifthereaxresuccessorsthenforeach one calledSUCCESSOR,thatisnotalso anancesterofNODEdothe following

(a) addSUCCESSORtographG

(b) ifsuccessor isnot aterminal node, mark itsolved and assign zeroto its h ' value.

(c) If successorisnot aterminalnode, computeith' value.

(III) propagatethenewlydiscoveredinformationupthegraphbydoingthefollowing.letSbea set of nodes that have been marked SOLVED. Initialize S to NODE. Until S is empty repeat thefollowingprocedure;

(a) selectanodefromS call ifCURRENTandremoveitfrom S.

(b) computeh'ofeachofthearcsemergingfromCURRENT,Assignminimumh'to CURRENT.

(c) Marktheminimum cost path a s thebest out of CURRENT.

(d) MarkCURRENTSOLVEDifallofthenodesconnectedtoitthroughthenewmarked are have been labeled SOLVED.

(e) If CURRENThas beenmarkedSOLVED oritsh'hasjustchanged,itsnew status must bepropagatebackwardsupthegraph.hencealltheancestorsofCURRENTareadded to S.

(ReferedFromArtificialIntelligenceTMH)

AO*Search Procedure.

1. Placethestart nodeonopen.

2. Usingthesearch tree,computethe mostpromisingsolution treeTP .

3. Selectnoden thatisboth onopenand apartoftp,removen fromopenandplaceitno closed.

4. Ifnisagoalnode,labelnassolved.Ifthestartnodeissolved,exitwithsuccesswheretpis the solution tree, remove all nodes from open with a solved ancestor.

5. Ifnisnotsolvablenode,labelnasunsolvable. Ifthestartnodeislabeledasunsolvable,exit with failure. Remove all nodes from open ,with unsolvable ancestors.

6. Otherwise,expandnodengeneratingallofitssuccessorcomputethecostofforeachnewly generated node and place all such nodes on open.

7. Gobacktostep(2)

Note:AO* will alwaysfind minimum cost solution.

CONSTRAINTSATISFACTION:-

ManyproblemsinAIcanbeconsideredasproblemsofconstraintsatisfaction,inwhichthegoal state satisfies a given set of constraint. constraint satisfaction problems can be solved by using any of the search strategies. The general form of the constraint satisfaction procedure is as follows:

Untilacompletesolution is foundoruntilall paths haveled toleadends, do

1. selectanunexpanded nodeofthesearchgraph.

2. Applytheconstraintinferencerulestotheselectednodetogenerateallpossiblenew constraints.

3. Ifthesetof constraintscontainsacontradiction,thenreportthatthispathisadeadend.

4. Ifthesetof constraintsdescribesa completesolutionthenreportsuccess.

5. Ifneitheraconstraintnoracompletesolutionhasbeenfoundthenapplytherulestogenerate new partial solutions. Insert these partial solutions into the search graph.

Example:considerthecryptarithmeticproblems.

```
   SEND
  +MORE
-------

MONEY
-------
```

Assigndecimaldigittoeachofthelettersinsuch awaythattheanswertotheproblemiscorrect to the same letter occurs more than once , it must be assign the same digit each time . no two different letters may be assigned the same digit. Consider the crypt arithmetic problem.

```
  SEND
 +MORE
-------
MONEY
-------
```

CONSTRAINTS:-

1. notwodigit canbeassigned tosame letter.

2. onlysingle digit numbercan beassignto a letter.

1. notwoletters canbeassignedsamedigit.

2. Assumptioncan bemadeatvarious levels suchthat theydo notcontradict each other.

3. Theproblemcanbedecomposedintosecuredconstraints.Aconstraintsatisfactionapproach may be used.

4. Anyof search techniques maybe used.

5. Backtrackingmaybe performedasapplicable usappliedsearch techniques.

6. Ruleofarithmeticmaybefollowed.

Initialstateofproblem.
D=?
E=?
Y=?
N=?
R=?
O=?
S=?
M=?
C1=?
C2=?
C1,C 2, C3 stands forthe carryvariables respectively.

GoalState: thedigitsto thelettersmust be assignedin such amanner so thatthesum is satisfied.

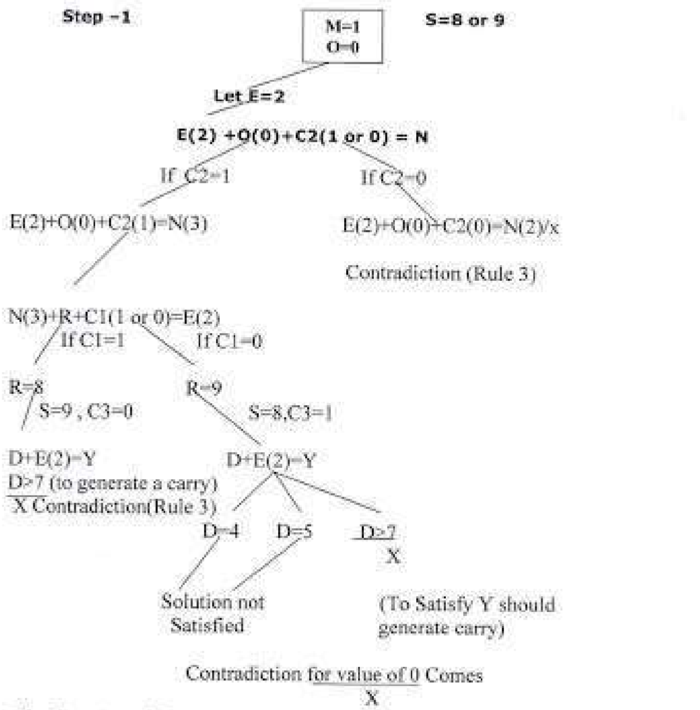
Solution Process:

Wearefollowingthe depth-first method to solvetheproblem.

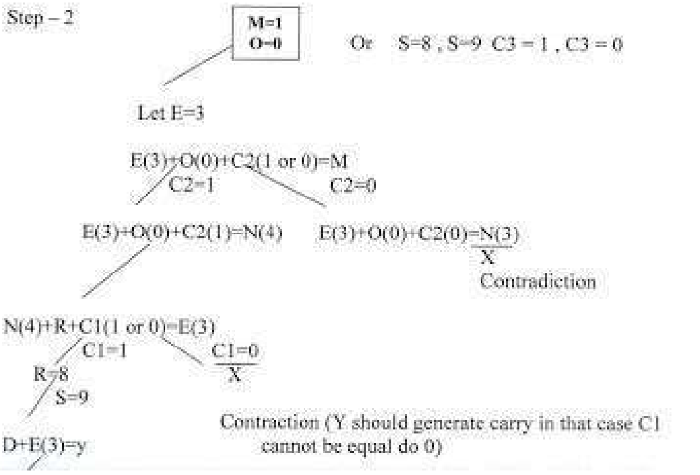
1. initialguessm=1becausethesum oftwosingle digitscangenerateatmostacarry'1'.
2. When n=1 o=0 or 1 because the largest single digit number added to m=1 can generate the sumofeither0or1dependonthecarryreceivedfromthecarrysum.Bythisweconcludethat o=0 because m is already 1 hence we cannot assign same digit another letter(rule no.)

3. Wehavem=1ando=0togeto=0wehaves=8or9,againdependingonthecarryreceived from the earlier sum.

The same process can be repeated further. The problem has to be composed into various constraints.Andeachconstraintsistobesatisfiedbyguessingthepossibledigitsthattheletters can be assumed that the initial guess has been alreadymade. rest of the process is beingshown in the form of a tree, using depth-first search for the clear understandability of the solution process.



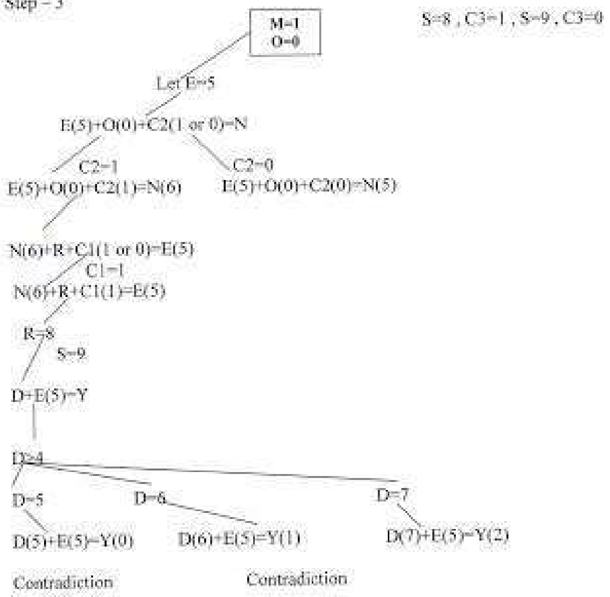
After Step 1 we derive are more conclusion that Y contradiction should generate a Carry. That is D+2>9



D>6(Controduction)

After Step 2 , we found that C1 cannot be Zero, Since Y has to generate a carry to satisfy goal state. From this step onwards, no need to branch for C1=0.

Step – 3



At Step (4) we have assigned a single digit to every letter in accordance with the constraints & production rules.

Now by backtracking , we find the different digits assigned to different letters and hence reach the solution state.

Solution State:-

Y = 2
D = 7
S = 9
R = 8
N = 6
E = 5
O = 0
M = 1
C1 = 1
C2 = 0
C3 = 0

```
          C3(0)  C2(1)  C1(1)
          S(9)   E(5)   N(6)   D(7)
      +   M(1)   O(0)   R(8)   E(5)
  ------------------------------------
   M(1)   O(0)   N(6)   E(5)   Y(2)
  ------------------------------------
```

MEANS-ENDSANALYSIS:-

Mostofthesearchstrategieseitherreasonforwardofbackwardhowever,oftena mixtureothe two directions is appropriate. Such mixed strategy would make it possible to solve the major parts of problem first and solve the smaller problems the arise when combiningthem together. Such a technique is called "Means - Ends Analysis".

Themeans-endsanalysisprocesscentersaroundfindingthedifferencebetweencurrentstateand goal state. The problem space of means - ends analysis has an initial state and one or more goal state, a set of operate with a set of preconditions their application and difference functions that computes the difference between two state a(i) and s(j). A problem is solved using means - ends analysis by

1. Computingthecurrent states1to agoalstate s2 andcomputingtheirdifferenceD12.

2. Satisfythepreconditionsforsomerecommendedoperatoropisselected,thentoreducethe difference D12.

3. TheoperatorOPisappliedifpossible.Ifnotthecurrentstateissolvedagoaliscreatedand means-ends analysis is applied recursively to reduce the sub goal.

4. Ifthesubgoalissolvedstateisrestoredandworkresumedontheoriginal problem.

(thefirstAIprogramtousemeans-endsanalysis wasthe GPSGeneral problem solver)

means-endsanalysis Iusefulformanyhumanplanningactivities.Considertheexampleof planing for an office worker. Suppose we have a different table of three rules:

1. Ifinoutcurrentstatewearehungry,andinourgoalstatewearenothungry,theneitherthe "visit hotel" or "visit Canteen " operator is recommended.

2. Ifourcurrentstatewedonothavemoney,andifinyourgoalstatewehavemoney,thenthe "Visit our bank" operator or the "Visit secretary" operator is recommended.

3. Ifourcurrentstatewedonotknowwheresomethingis,needinourgoalstatewedoknow, then either the "visit office enquiry" , "visit secretary" or "visit co worker " operator is recommended.

KNOWLEDGEREPRESENTATION

KNOWLEDGEREPRESENTATION:-

Forthepurposeofsolvingcomplexproblemsc\encounteredinAI,weneedbothalargeamount of knowledge and some mechanism for manipulating that knowledge to create solutions to new problems. A variety of ways of representing knowledge (facts) have been exploited in AI programs. In all variety of knowledge representations , we deal with two kinds of entities.

A. Facts:Truthsinsomerelevantworld.These arethethingswewantto represent.

B. Representationsoffactsinsomechosenformalism.thesearethingswewill actually be able to manipulate.
Onewaytothinkofstructuringtheseentitiesisattwolevels:(a)theknowledgelevel, atwhich facts are described, and (b) the symbol level, at which representations of objects at the knowledge level are defined in terms of symbols that can be manipulated by programs.

The facts and representations are linked with two-way mappings. This link is called representation mappings. The forward representation mapping maps from facts to representations.Thebackwardrepresentationmappinggoestheotherway,fromrepresentations to facts.

One common representation is natural language (particularly English) sentences. Regardless of the representation for facts we use in a program , we may also need to be concerned with an English representation of those facts in order to facilitate getting information into and out of the system.WeneedmappingfunctionsfromEnglishsentencestotherepresentationweactuallyuse and from it back to sentences.

RepresentationsandMappings

- Inordertosolvecomplexproblemsencounteredinartificialintelligence,oneneedsboth a large amount of knowledge and some mechanism for manipulating that knowledge to create solutions.
- KnowledgeandRepresentationaretwodistinctentities.Theyplaycentralbut distinguishable roles in the intelligent system.
- Knowledgeisadescriptionoftheworld.Itdeterminesasystem'scompetencebywhatit knows.
- Moreover,Representationisthewayknowledgeisencoded.Itdefinesasystem's performance in doing something.
- Differenttypes of knowledgerequiredifferent kinds of representation.

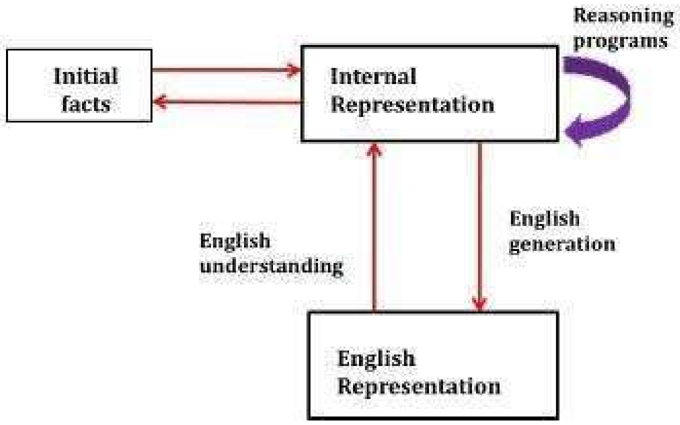


Fig:MappingbetweenFactsandRepresentations

TheKnowledgeRepresentationmodels/mechanisms areoftenbased on:

- Logic
- Rules
- Frames
- SemanticNet

Knowledgeiscategorized intotwomajortypes:

1. Tacitcorrespondsto“informal”or“implicit“
    - Existswithin ahuman being;
    - Itis embodied.
    - Difficulttoarticulate formally.
    - Difficulttocommunicate orshare.
    - Moreover,Hard tostealorcopy.
    - Drawnfromexperience,action,subjective insight
2. Explicitformaltypeofknowledge,Explicit
    - Explicit knowledge
    - Existsoutsideahuman being;
    - Itis embedded.
    - Canbearticulatedformally.
    - Also,Canbeshared,copied,processed andstored.
    - So,Easytosteal orcopy
    - Drawnfromtheartifactofsometypeasaprinciple,procedure,process,concepts. A

variety of ways of representing knowledge have been exploited in AI programs.

Therearetwo differentkinds of entities,wearedealingwith.

1. Facts:Truthinsomerelevantworld.Thingswewantto represent.
2. Also,Representationoffactsinsomechosenformalism.Thingswewillactuallybeable to manipulate.

Theseentitiesstructured attwolevels:

1. Theknowledgelevel,atwhichfactsdescribed.
2. Moreover,Thesymbollevel,atwhichrepresentationofobjectsdefinedintermsof symbols that can manipulate by programs

FrameworkofKnowledgeRepresentation

- Thecomputerrequiresawell-definedproblemdescriptiontoprocessandprovideawell-defined acceptable solution.

- Moreover,Tocollectfragmentsofknowledgeweneedfirsttoformulateadescriptionin our spoken language and then represent it in formal language so that computer can understand.
- Also,Thecomputercanthenuseanalgorithmtocomputeananswer. So,

This process illustrated as,

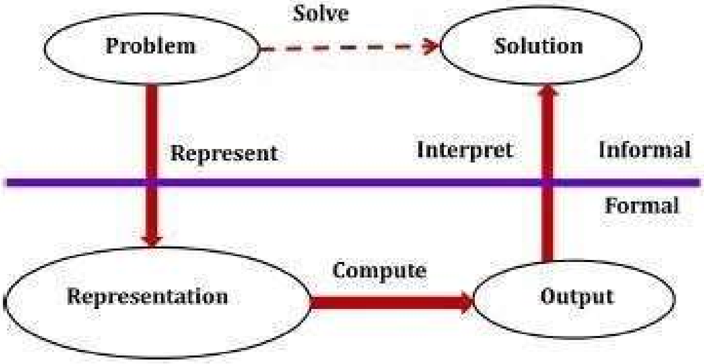


Fig:KnowledgeRepresentation Framework

Thesteps are:

- Theinformal formalismof theproblem takesplacefirst.
- Itthenrepresentedformallyandthe computerproducesanoutput.
- Thisoutputcanthenrepresentedinaninformallydescribedsolutionthatuserunderstands or checks for consistency.

TheProblemsolving requires,

- Formalknowledgerepresentation,and
- Moreover,Conversionofinformalknowledgetoaformalknowledgethatisthe conversion of implicit knowledge to explicit knowledge.

MappingbetweenFactsandRepresentation

- Knowledgeisa collectionoffactsfromsome domain.
- Also,Weneed arepresentation of “facts“that can manipulatebyaprogram.
- Moreover,NormalEnglishisinsufficient,toohardcurrentlyforacomputerprogramto draw inferences in natural languages.
- Thussomesymbolicrepresentationisnecessary.

Agoodknowledgerepresentationenablesfastandaccurateaccesstoknowledgeand understanding of the content.

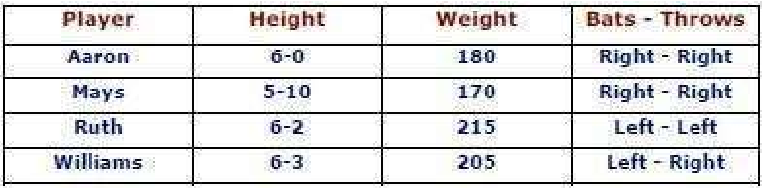
Aknowledgerepresentationsystem shouldhavefollowingproperties.

1. RepresentationalAdequacy
   - Theabilityto representall kinds ofknowledgethat areneeded in that domain.
2. InferentialAdequacy
   - Also,Theabilitytomanipulatetherepresentationalstructurestoderivenew structures corresponding to new knowledge inferred from old.
3. InferentialEfficiency
   - Theabilitytoincorporateadditionalinformationintotheknowledgestructurethat can be used to focus the attention of the inference mechanisms in the most promising direction.
4. AcquisitionalEfficiency
   - Moreover,Theabilitytoacquirenewknowledgeusingautomaticmethods wherever possible rather than reliance on human intervention.

KnowledgeRepresentationSchemes

Relational Knowledge

- Thesimplestwaytorepresentdeclarativefactsisasetofrelationsofthesamesortused in the database system.
- Provides a framework to compare two objects based on equivalent attributes. o Any instanceinwhichtwodifferentobjectsarecomparedisarelationaltypeofknowledge.
- Thetablebelow showsasimple wayto store facts.
  - Also,Thefactsabout a set ofobjects areput systematicallyin columns.
  - Thisrepresentationprovideslittleopportunityfor inference.

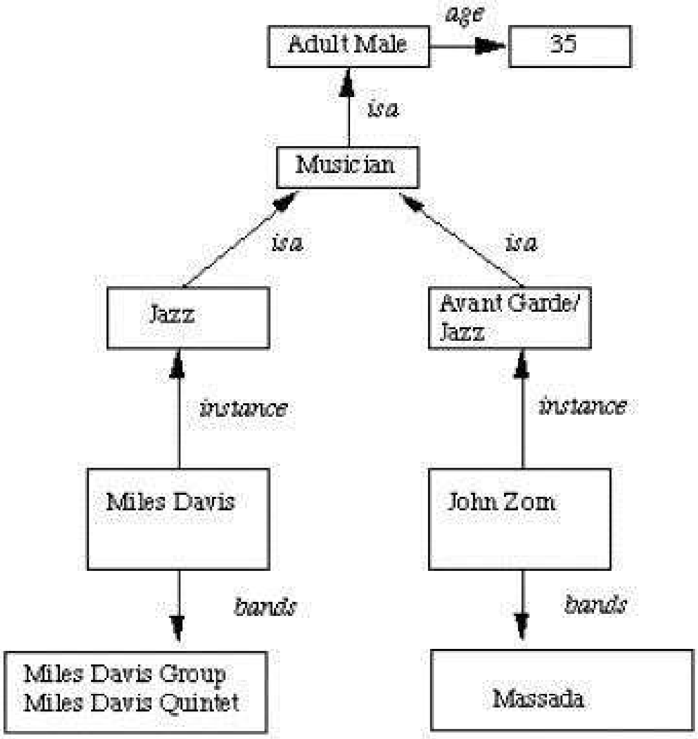


Player	Height	Weight	Bats - Throws
Aaron	6-0	180	Right - Right
Mays	5-10	170	Right - Right
Ruth	6-2	215	Left - Left
Williams	6-3	205	Left - Right

- Giventhefacts,itisnotpossibletoanswerasimplequestionsuchas:"Whoisthe heaviest player?"
- Also,Butifaprocedureforfindingtheheaviestplayerisprovided,thenthesefactswill enable that procedure to compute an answer.
- Moreover,Wecan askthingslikewho"bats –left"and"throws–right".

Inheritable Knowledge

- Heretheknowledge elementsinheritattributesfromtheir parents.
- Theknowledgeembodiedinthedesignhierarchiesfoundinthefunctional,physicaland process domains.
- Withinthehierarchy,elementsinheritattributesfromtheirparents,butinmanycases,not all attributes of the parent elements prescribed to the child elements.
- Also,Theinheritanceisa powerfulform ofinference, butnot adequate.
- Moreover,ThebasicKR(KnowledgeRepresentation)needstoaugmentwithinference mechanism.
- Propertyinheritance:Theobjectsorelementsofspecificclassesinheritattributesand values from more general classes.
- So,Theclassesorganized inageneralized hierarchy.



- Boxednodes— objectsandvalues ofattributesofobjects.
- Arrows—thepointfromobjecttoits value.
- Thisstructureisknownasaslotandfillerstructure,semanticnetworkoracollectionof frames.

Thestepsto retrieveavalueforanattributeof aninstanceobject:

1. Findtheobject intheknowledgebase
2. Ifthereisavaluefortheattributereportit
3. Otherwiselookforavalueofaninstance, ifnone fail
4. Also,Go to thatnode and findavalue fortheattribute andthen report it
5. Otherwise,search throughusingisuntil avalue isfound forthe attribute.

Inferential Knowledge

- Thisknowledgegeneratesnewinformationfromthegiven information.
- Thisnewinformationdoesnotrequirefurtherdatagatheringformsourcebutdoes require analysis of the given information to generate new knowledge.
- Example:givenasetofrelationsandvalues,onemayinferothervaluesorrelations.A predicate logic (a mathematical deduction) used to infer from a set of attributes. Moreover, Inference through predicate logic uses a set of logical operations to relate individual data.
- Representknowledgeasformallogic:Alldogs havetails∀x*:dog(x)→ hastail(x)*
- Advantages:
  - Asetofstrict rules.
  - Canusetoderive morefacts.
  - Also,Truthsofnew statementscanbeverified.
  - Guaranteedcorrectness.
- So,Manyinferenceproceduresavailabletoimplementstandardrulesoflogicpopularin AI systems. e.g Automated theorem proving.

ProceduralKnowledge

- Arepresentationinwhichthecontrolinformation,tousetheknowledge,embeddedinthe knowledgeitself.Forexample,computerprograms,directions,andrecipes;theseindicate specific use or implementation;
- Moreover,Knowledgeencodedinsomeprocedures,smallprogramsthatknowhowtodo specific things, how to proceed.
- Advantages:
    - Heuristicordomain-specificknowledgecan represent.
    - Moreover,Extendedlogicalinferences,suchasdefaultreasoningfacilitated.
    - Also,Sideeffectsofactionsmaymodel.Somerulesmaybecomefalseintime. Keeping track of this in large systems may be tricky.
- Disadvantages:
    - Completeness — notall cases mayrepresent.
    - Consistency—notalldeductionsmaybecorrect. e.g IfweknowthatFred isa birdwemightdeducethatFredcanfly.LaterwemightdiscoverthatFredisan emu.
    - Modularitysacrificed.Changesinknowledgebasemighthavefar-reaching effects.
    - Cumbersomecontrol information.

USINGPREDICATELOGIC

RepresentationofSimpleFactsinLogic

Propositionallogicisusefulbecauseitissimpletodealwithandadecisionprocedureforit exists. Also,Inorder todraw conclusions,factsarerepresentedin amore convenientway as,

1. Marcusisaman.
    - man(Marcus)
2. Plato is aman.
    - man(Plato)
3. Allmenare mortal.
    - mortal(men)

Butpropositionallogicfailstocapturetherelationshipbetweenanindividualbeingamanand that individual being a mortal.

- Howcanthesesentencesberepresentedsothatwecaninferthethirdsentencefromthe first two?
- Also,Propositionallogiccommitsonlytotheexistenceoffactsthatmayormaynotbe the case in the world being represented.
- Moreover,Ithasasimplesyntaxandsimplesemantics.Itsufficestoillustratetheprocess of inference.
- Propositionallogic quicklybecomes impractical, even forverysmall worlds.

Predicatelogic

First-orderPredicatelogic(FOPL)modelstheworldintermsof

- Objects,whicharethingswithindividual identities
- Propertiesofobjectsthat distinguishthemfromotherobjects
- Relationsthat hold amongsets of objects

- Functions,whichareasubsetofrelationswherethereisonlyone“value”foranygiven “input”

First-orderPredicatelogic(FOPL) provides

- Constants:a,b,dog33. Nameaspecificobject.
- Variables:X,Y.Refertoanobjectwithoutnamingit.
- Functions:Mappingfromobjectstoobjects.
- Terms:Refertoobjects
- AtomicSentences:in(dad-of(X),food6)Canbetrueorfalse,Correspondtopropositional symbols P, Q.

Awell-formedformula(*wff*)isasentencecontainingno“free”variables.So,Thatis,all variables are “bound” by universal or existential quantifiers.

(∀x)P(x,y)has xboundas auniversallyquantifiedvariable,butyis free.

Quantifiers

Universalquantification

- (∀x)P(x)means thatP holdsforallvalues ofxinthedomainassociatedwiththat variable
- E.g.,(∀x)dolphin(x)→mammal(x)

Existential quantification

- (∃ x)P(x)meansthatPholdsforsomevalueofxinthedomainassociatedwiththat variable
- E.g.,(∃x)mammal(x)∧lays-eggs(x)

Also,Considerthefollowingexamplethatshows theuseofpredicatelogicasawayof representing knowledge.

1. Marcuswasaman.
2. MarcuswasaPompeian.
3. AllPompeianswere Romans.
4. Caesarwasa ruler.
5. Also,AllPompeians wereeither loyaltoCaesar orhated him.
6. Everyoneisloyaltosomeone.
7. Peopleonlytryto assassinate rulerstheyarenot loyalto.
8. Marcustriedtoassassinate Caesar.

Thefactsdescribedbythesesentencescanberepresentedas asetofwell-formedformulas(*wffs*) as follows:

1. Marcuswasaman.
    - man(Marcus)
2. MarcuswasaPompeian.
    - Pompeian(Marcus)
3. AllPompeianswere Romans.
    - ∀x:Pompeian(x)→Roman(x)
4. Caesarwasa ruler.
    - ruler(Caesar)
5. AllPompeianswereeitherloyalto Caesarorhated him.
    - inclusive-or
    - ∀x:Roman(x) →loyalto(x, Caesar)∨hate(x, Caesar)
    - exclusive-or
    - ∀x:Roman(x)→(loyalto(x, Caesar)∧¬hate(x,Caesar))∨
    - (¬loyalto(x,Caesar)∧hate(x, Caesar))

6. Everyoneisloyaltosomeone.
    - ∀x:∃y:loyalto(x,y)
7. Peopleonlytryto assassinate rulerstheyarenot loyalto.
    - ∀x:∀y:person(x)∧ruler(y)∧tryassassinate(x,y)
    - →¬loyalto(x,y)
8. Marcustriedtoassassinate Caesar.
    - tryassassinate(Marcus,Caesar)

Nowsupposeifwewanttousethesestatementstoanswerthequestion: ***WasMarcusloyalto Caesar?*** Also,Nowlet'strytoproduceaformalproof,reasoningbackwardfromthedesiredgoal:¬ Ioyalto(Marcus, Caesar)

In order to prove the goal, we need to use the rules of inference to transform it into another goal (orpossiblyasetofgoals)thatcan,inturn,transformed,andsoon, untiltherearenounsatisfied goals remaining.

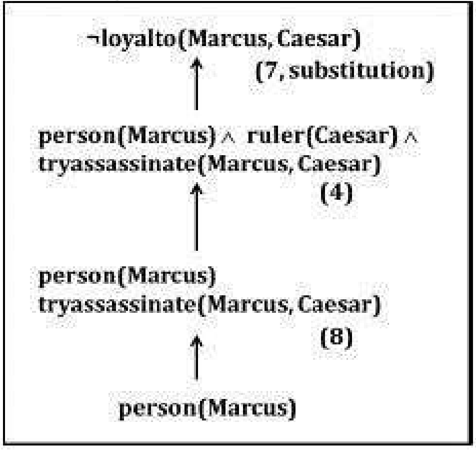
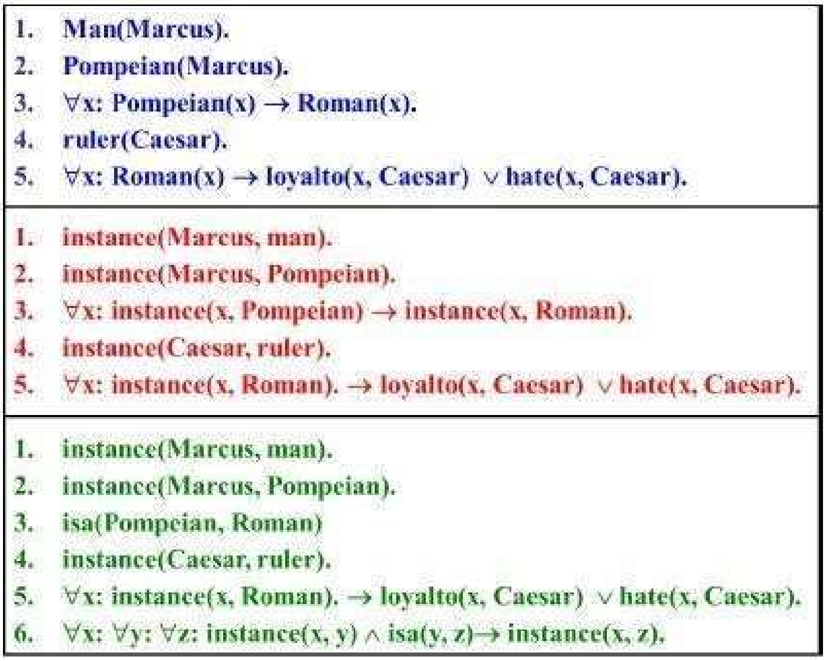

¬loyalto(Marcus, Caesar)
(7, substitution)
person(Marcus) ∧ ruler(Caesar) ∧
tryassassinate(Marcus, Caesar)
(4)
person(Marcus)
tryassassinate(Marcus, Caesar)
(8)
person(Marcus)

Figure:An attempttoprove¬loyalto(Marcus,Caesar).

- Theproblem is that, although weknow that Marcus was aman, wedo not haveanyway toconcludefromthatthatMarcuswasaperson.Also,Weneedtoaddtherepresentation ofanotherfacttooursystem,namely:***∀man(x)→person(x)***
- Nowwecansatisfythelast goaland produceaproof thatMarcus wasnot loyal to Caesar.
- Moreover, From this simple example, we see that three important issues must be addressedintheprocessofconvertingEnglishsentencesintologicalstatementsandthen using those statements to deduce new ones:
    1. ManyEnglishsentencesareambiguous(forexample,5,6,and7above). Choosing the correct interpretation may be difficult.
    2. Also, There is often a choice of how to represent the knowledge. Simple representationsaredesirable,buttheymayexcludecertainkindsofreasoning.
    3. Similalry, Even in verysimple situations, a set of sentences is unlikelyto contain alltheinformationnecessarytoreasonaboutthetopicathand.Inordertobeable to use a set of statements effectively. Moreover, It is usually necessary to have access to another set of statements that represent facts that people consider too obvious to mention.

RepresentingInstanceandISARelationships

- Specificattributes**instance**and**isa**playanimportantroleparticularlyin ausefulformof reasoning called property inheritance.
- Thepredicatesinstanceandisaexplicitlycapturedtherelationshipstheyusedtoexpress, namely class membership and class inclusion.
- 4.2showsthefirstfivesentencesofthelastsectionrepresentedinlogicinthreedifferent ways.
- Thefirstpartofthefigurecontainstherepresentationswehavealreadydiscussed.In these representations, class membership represented with unary predicates (such as Roman), each of which corresponds to a class.
- Assertingthat P(x)is trueis equivalent toassertingthatxis aninstance(orelement) ofP.
- Thesecondpartofthefigurecontainsrepresentationsthatusethe***instance***predicate explicitly.



1. Man(Marcus).
2. Pompeian(Marcus).
3. ∀x: Pompeian(x) → Roman(x).
4. ruler(Caesar).
5. ∀x: Roman(x) → loyalto(x, Caesar) ∨ hate(x, Caesar).

1. instance(Marcus, man).
2. instance(Marcus, Pompeian).
3. ∀x: instance(x, Pompeian) → instance(x, Roman).
4. instance(Caesar, ruler).
5. ∀x: instance(x, Roman). → loyalto(x, Caesar) ∨ hate(x, Caesar).

1. instance(Marcus, man).
2. instance(Marcus, Pompeian).
3. isa(Pompeian, Roman)
4. instance(Caesar, ruler).
5. ∀x: instance(x, Roman). → loyalto(x, Caesar) ∨ hate(x, Caesar).
6. ∀x: ∀y: ∀z: instance(x, y) ∧ isa(y, z)→ instance(x, z).

Figure:Threewaysof representingclass membership:ISARelationships

- Thepredicate***instance***isabinaryone,whosefirstargumentisanobjectandwhose second argument is a class to which the object belongs.
- Buttheserepresentations donot usean explicit***isa***predicate.
- Instead,subclassrelationships,suchasthatbetweenPompeiansandRomans,described as shown in sentence 3.
- TheimplicationrulestatesthatifanobjectisaninstanceofthesubclassPompeianthenit is an instance of the superclass Roman.
- Notethatthisruleisequivalenttothestandardset-theoreticdefinitionofthesubclass-superclass relationship.
- Thethirdpartcontainsrepresentationsthatuseboththe***instance***and***isa***predicates explicitly.
- Theuseofthe***isa***predicatesimplifiestherepresentationofsentence3,butitrequiresthat one additional axiom (shown here as number 6) be provided.

ComputableFunctionsandPredicates

- Toexpresssimplefacts,suchasthefollowinggreater-thanandless-thanrelationships: gt(1,O) It(0,1) gt(2,1)It(1,2) gt(3,2)It(2,3)
- Itisoftenalsousefultohavecomputablefunctionsaswellascomputablepredicates. Thus we might want to be able to evaluate the truth ofgt(2 + 3,1)
- Todosorequiresthatwefirstcomputethevalueoftheplusfunctiongiventhearguments 2 and 3, and then send the arguments 5 and 1 to gt.

Considerthefollowingset offacts, againinvolvingMarcus:

1) Marcuswasaman.
    man(Marcus)
2) MarcuswasaPompeian.
    Pompeian(Marcus)
3) Marcuswasbornin40A.D.
    born(Marcus, 40)
4) Allmenaremortal.
    x:man(x) → mortal(x)
5) All Pompeians died when the volcano erupted in 79 A.D.
    erupted(volcano,79)∧ ∀ x:[Pompeian(x)→died(x,79)]
6) Nomortallives longer than150years.
    x:t1:At2: *mortal(x)born(x,t1)gt(t2–t1,150)→died(x, t2)*
7) Itisnow 1991.
    now =1991

So,Aboveexampleshowshowtheseideasofcomputablefunctionsandpredicatescanbeuseful. It also makes use of the notion of equality and allows equal objects to be substituted for each other whenever it appears helpful to do so during a proof.

- So,Nowsupposewewantto answerthequestion"IsMarcusalive?"
- Thestatements suggestedhere, theremaybetwowaysof deducing an answer.
- Eitherwecanshowthat Marcusisdeadbecausehewaskilledbythevolcanoorwecan show that he must be dead because he would otherwise be more than 150 years old, which we know is not possible.
- Also, As soon as we attempt to follow either of those paths rigorously, however, we discover,justaswedidinthelastexample,thatweneedsomeadditionalknowledge.For example, our statements talk about dying, but theysaynothingthat relates to being alive, which is what the question is asking.

Soweadd the followingfacts:

8) Alivemeansnotdead.
    x:t: [alive(x, t)→ ¬dead(x, t)][¬dead(x, t) → alive(x, t)]
9) If someone dies, then he is dead at all later times.
    x:t1:At2*:died(x,t1)gt(t2,t1)→dead(x,t2)*

So,Nowlet'sattempttoanswerthequestion"IsMarcusalive?"byproving:¬***alive(Marcus, now)***

ResolutionPropositionalResolution

1. Convertall thepropositionsof Ftoclauseform.
2. NegatePandconverttheresulttoclauseform.Addittothesetofclausesobtainedin step 1.
3. Repeatuntil eitheracontradictionis found orno progresscan bemade:
    1. Selecttwoclauses.Callthesetheparentclauses.
    2. Resolve them together. The resulting clause, called the resolvent, will be the disjunction of all of the literals of both of the parent clauses with the following exception: IfthereareanypairsofliteralsLand ¬*L*suchthatoneoftheparent clauses contains *L* and the other contains ¬*L,* then select one such pair and eliminate both *L* and ¬ *L* from the resolvent.
    3. Iftheresolventistheemptyclause,thenacontradictionhasbeenfound.If itis not, then add it to the set of classes available to the procedure.

TheUnificationAlgorithm

- Inpropositionallogic,itiseasytodeterminethattwoliteralscannotbothbetrueatthe same time.
- SimplylookforLand¬Linpredicatelogic,thismatchingprocessismorecomplicated since the arguments of the predicates must be considered.
- Forexample,man(John)and¬man(John)isa contradiction,whiletheman(John)and ¬man(Spot)isnot.
- Thus,inordertodeterminecontradictions,weneedamatchingprocedurethatcompares two literals and discovers whether there exists a set of substitutions that makes them identical.
- Thereisastraightforwardrecursiveprocedure,calledtheunificationalgorithm,thatdoes it.

Algorithm:Unify(L1, L2)

1. IfL1or L2arebothvariablesorconstants,then:
    1. IfL1andL2 areidentical,thenreturnNIL.
    2. ElseifL1isavariable,thenifL1occursinL2thenreturn{FAIL},elsereturn (L2/L1).
    3. Also,ElseifL2isavariable,thenifL2occursinL1thenreturn{FAIL},else return (L1/L2). d. Else return {FAIL}.
2. IftheinitialpredicatesymbolsinL1andL2arenotidentical,thenreturn{FAIL}.
3. If LIandL2have adifferent numberofarguments,thenreturn {FAIL}.
4. SetSUBSTtoNIL.(Attheendofthisprocedure,SUBSTwillcontainallthe substitutions used to unify L1 and L2.)
5. ForI←1tothenumber ofargumentsinL1:
    1. CallUnifywiththeithargumentofL1andtheithargumentofL2,puttingthe result in S.
    2. If ScontainsFAILthenreturn {FAIL}.
    3. If Sis notequal toNILthen:
        2. ApplyStotheremainderofbothL1andL2.
        3. SUBST:=APPEND(S, SUBST).
6. ReturnSUBST.

ResolutioninPredicate Logic

Wecannowstatetheresolutionalgorithmforpredicatelogicasfollows,assumingasetofgiven statements F and a statement to be proved P:

Algorithm:Resolution

1. Convertall the statements ofFto clauseform.
2. NegatePand convert the resulttoclauseform.Addit tothesetof clausesobtainedin 1.
3. Repeatuntilacontradictionfound,noprogresscanmake,orapredeterminedamountof effort has expanded.
    1. Selecttwoclauses.Callthesetheparentclauses.
    2. Resolvethemtogether.Theresolventwillthedisjunctionofalltheliteralsofboth parent clauses with appropriate substitutions performed and with the following exception: If there is one pair of literals T1 and ¬T2 such that one of the parent clausescontainsT2 andtheothercontainsT1 and ifT1andT2areunifiable,then neither T1 nor T2 should appear in the resolvent. We call T1 and T2 Complementaryliterals.Usethesubstitutionproducedbytheunificationtocreate the resolvent. If there is more than one pair of complementary literals, only one pair should omit from the resolvent.
    3. Iftheresolventisanemptyclause,thenacontradictionhasfound.Moreover, Ifit is not, then add it to the set of classes available to the procedure.

ResolutionProcedure

- Resolutionisaprocedure,whichgainsitsefficiencyfromthefactthatitoperateson statements that have been converted to a very convenient standard form.
- Resolutionproducesproofs byrefutation.
- Inotherwords,***toproveastatement(i.e.,toshowthatitisvalid),resolutionattemptsto show that the negation of the statement produces a contradiction with the known statements (i.e., that it is unsatisfiable).***
- The resolution procedure is a simple iterative process: at each step, two clauses, called the parent clauses, are compared (resolved), resulting in a new clause that has inferred fromthem.Thenewclauserepresentswaysthatthetwoparentclausesinteractwitheach other. Suppose that there are two clauses in the system:

winter* V *summer

¬winter* V *cold

- Nowweobservethatpreciselyoneof winterand¬winterwillbetrue at anypoint.
- Ifwinteristrue,thencoldmustbetruetoguaranteethetruthofthesecondclause.If¬ winter is true, then summer must be true to guarantee the truth of the first clause.
- Thusweseethat fromthesetwo clauses we candeduce***summer Vcold***
- Thisisthedeductionthat theresolutionprocedurewill make.
- Resolutionoperatesbytakingtwoclausesthateachcontainsthesameliteral,inthis example, ***winter***.
- Moreover,Theliteralmustoccurinthepositiveforminoneclauseandinnegativeform in the other. The resolvent obtained by combining all of the literals of the two parent clauses except the ones that cancel.
- If the clausethat produced istheemptyclause, then acontradiction hasfound.

Forexample,thetwoclauses

winter

¬winter
will producethe emptyclause.

NaturalDeductionUsing Rules

Testing whether a proposition is a tautology by testing every possible truth assignment is expensive—thereareexponentiallymany.Weneeda**deductivesystem**,whichwillallowusto construct proofs of tautologies in a step-by-step fashion.

Thesystem wewilluseisknownas **naturaldeduction**.Thesystemconsistsofasetof **rulesof inference** for deriving consequences from premises. One builds a proof tree whose root is the propositiontobeprovedandwhoseleavesaretheinitialassumptionsoraxioms(forprooftrees, we usually draw the root at the bottom and the leaves at the top).

Forexample,oneruleofoursystemisknownas **modusponens**.Intuitively,thissaysthatifwe know P is true, and we know that P implies Q, then we can conclude Q.

$$\frac{P \quad P \Rightarrow Q}{Q}\,(\text{modusponens})$$

Thepropositions abovethelineare called**premises**; thepropositionbelow thelineis the**conclusion**.Boththepremisesandtheconclusionmaycontainmetavariables(inthiscase,P andQ)representingarbitrarypropositions.Whenaninferenceruleisusedaspartofaproof,the metavariables are replaced in a consistent way with the appropriate kind of object (in this case, propositions).

Most rules come in one of two flavors: **introduction** or **elimination** rules. Introduction rules introduce the use of a logical operator, and elimination rules eliminate it. Modus ponens is an eliminationrulefor⇒.Ontheright-handsideofarule,weoftenwritethenameoftherule.This is helpful when reading proofs. In this case, we have written (modus ponens). We could also have written (⇒-elim) to indicate that this is the elimination rule for ⇒.

Rules forConjunction

Conjunction(∧)hasanintroductionruleandtwoelimination rules:

$$\frac{P \qquad Q}{P \wedge Q}\,(\wedge\text{-intro}) \qquad \frac{P \wedge Q}{P}\,(\wedge\text{-elim-left}) \qquad \frac{P \wedge Q}{Q}\,(\wedge\text{-elim-right})$$

Rulefor T

ThesimplestintroductionruleistheoneforT. It iscalled"unit".Becauseithasnopremises,this rule is an **axiom**: something that can start a proof.

$$\frac{}{T}\,(\text{unit})$$

Rules forImplication

Innaturaldeduction,toproveanimplicationoftheformP ⇒ Q,weassumeP,thenreasonunder that assumption to tryto derive Q. If we are successful, then we can conclude that P ⇒ Q.

In a proof, we are always allowed to introduce a new assumption P, then reason under that assumption.Wemustgivetheassumptionaname;wehaveusedthenamexintheexample below. Each distinct assumption must have a different name.

$$\frac{}{[x: P]}\,(\text{assum})$$

Becauseithasnopremises,thisrulecanalsostartaproof.ItcanbeusedasifthepropositionP were proved. The name of the assumption is also indicated here.

However, youdonotgettomakeassumptionsforfree!Togetacompleteproof,allassumptions must be eventually *discharged*. This is done in the implication introduction rule. This rule introduces an implication P ⇒ Q bydischarging a prior assumption [x : P]. Intuitively, if Q can be proved under the assumption P, then the implication P ⇒ Q holds without any assumptions. We write x in the rule name to show which assumption is discharged. This rule and modus ponens are the introduction and elimination rules for implications.

$$\frac{\begin{array}{c}[x: P]\\ \vdots \\ Q\end{array}}{P \Rightarrow Q}\ (\Rightarrow\text{-intro/x}) \qquad \frac{P \quad P \Rightarrow Q}{Q}\ (\Rightarrow\text{-elim,modusponens})$$

Aproofisvalidonlyifeveryassumptioniseventuallydischarged.Thismusthappenintheproof tree below the assumption. The same assumption can be used more than once.

Rules forDisjunction

$$\frac{P}{P \vee Q}\ (\vee\text{-intro-left}) \qquad \frac{Q}{P \vee Q}\ (\vee\text{-intro-right}) \qquad \frac{P \vee Q \qquad P \Rightarrow R \qquad Q \Rightarrow R}{R}\ (\vee\text{-elim})$$

RulesforNegation

Anegation¬PcanbeconsideredanabbreviationforP⇒⊥:

$$\frac{P \Rightarrow \bot}{\neg P}\ (\neg\text{-intro}) \qquad \frac{\neg P}{P \Rightarrow \bot}\ (\neg\text{-elim})$$

Rules forFalsity

$$\frac{\begin{array}{c}[x: \neg P]\\ \vdots \\ \bot\end{array}}{P}\ (\text{reductioadabsurdum,RAA/x}) \qquad \frac{\bot}{P}\ (\text{exfalsoquodlibet,EFQ})$$

Reductioadabsurdum(RAA)isaninterestingrule.Itembodiesproofsbycontradiction.Itsays that if by assuming that P is false we can derive a contradiction, then P must be true. The assumption x is discharged in the application of this rule. This rule is present in classical logic but not in **intuitionistic** (constructive) logic. In intuitionistic logic, a proposition is not considered true simply because its negation is false.

ExcludedMiddle

Anotherclassicaltautologythatisnotintuitionisticallyvalidisthe **thelawoftheexcluded middle**, P ∨ ¬P. We will take it as an axiom in our system. The Latin name for this rule is*tertium non datur*, butwewill call it*magic*.

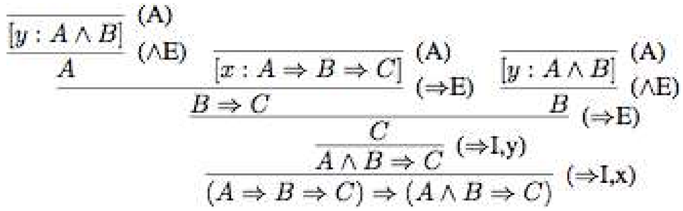
$$\frac{}{P \vee \neg P}\ (\text{magic})$$

Proofs

AproofofpropositionPinnaturaldeductionstartsfromaxiomsandassumptionsandderivesP withallassumptionsdischarged.Everystepinthe proofisaninstanceofan inferencerulewith metavariables substituted consistently with expressions of the appropriate syntactic class.

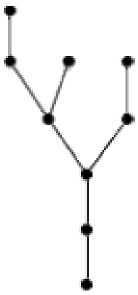
Example

Forexample, hereisa proof oftheproposition(A ⇒B⇒C)⇒(A∧B⇒C).



Thefinal step in theproof is to derive (A⇒ B⇒ C)⇒ (A∧ B⇒ C)from (A∧ B⇒ C), which is done usingthe rule (⇒-intro), dischargingthe assumption [x : A ⇒ B ⇒ C].To see how this rule generates the proof step, substitute for the metavariables P, Q, x in the rule as follows: P = (A ⇒ B⇒ C),Q=(A∧ B⇒ C),andx =x.Theimmediatelypreviousstepusesthesamerule,butwith a different substitution: P = A ∧ B, Q = C, x = y.

Theprooftreeforthisexamplehasthefollowingform,withtheprovedpropositionattheroot and axioms and assumptions at the leaves.



Apropositionthat hasacompleteproof inadeductivesystem iscalled a**theorem**ofthat system.

SoundnessandCompleteness

A measure of a deductive system's power is whether it is powerful enough to prove all true statements.Adeductivesystemissaidtobe**complete**ifalltruestatementsaretheorems(have proofs in the system). For propositional logic and natural deduction, this means that all tautologies must have natural deduction proofs. Conversely, a deductive system is called **sound** if all theorems aretrue. Theproof rules wehave given above arein fact sound and completeforpropositionallogic:everytheoremisatautology,andeverytautologyisa theorem.

Findingaproof fora given tautologycan bedifficult. But oncethe proof is found, checkingthat it is indeed a proof is completelymechanical, requiring no intelligence or insight whatsoever. It is therefore a very strong argument that the thing proved is in fact true.

We can also make writing proofs less tedious by adding more rules that provide reasoning shortcuts.Theserules aresoundifthereisawaytoconvertaproofusingthemintoaproofusing the original rules. Such added rules are called **admissible**.

ProceduralversusDeclarativeKnowledge

Wehavediscussedvarioussearchtechniquesinpreviousunits.Nowwewouldconsiderasetof rules that represent,

1. Knowledgeaboutrelationshipsintheworld and
2. Knowledgeabouthow tosolvethe problem usingthe contentofthe rules.

ProceduralvsDeclarativeKnowledge

Procedural Knowledge

- Arepresentationinwhichthecontrolinformationthatisnecessarytousetheknowledge isembeddedintheknowledgeitselffore.g.computerprograms,directions,andrecipes; theseindicate specific use or implementation;
- Therealdifferencebetweendeclarativeandproceduralviewsofknowledgeliesinwhere control information reside.

Forexample,considerthe following

Man (Marcus)Man
(Caesar)
Person(Cleopatra)
$\forall$*x:Man(x)→Person(x)*
Now,trytoanswer thequestion.?Person(y)

Theknowledgebasejustifies anyof the followinganswers.

Y=Marcus
Y=Caesar
Y=Cleopatra

- Weget morethan onevalue thatsatisfies the predicate.
- Ifonlyonevalueneeded,thentheanswertothequestionwilldependontheorderin which the assertions examined during the search for a response.
- Iftheassertionsdeclarativethentheydonotthemselvessayanythingabouthowtheywill be examined. In case of procedural representation, they say how they will examine.

DeclarativeKnowledge

- Astatementinwhichknowledgespecified,buttheusetowhichthatknowledgeistobe put is not given.
- Forexample,laws,people'sname;thesearethefactswhichcanstandalone,not dependent on other knowledge;
- Sotousedeclarativerepresentation,wemusthaveaprogramthatexplainswhatistodo with the knowledge and how.
- Forexample,asetoflogicalassertionscancombinewitharesolutiontheoremproverto give a complete program for solving problems but in some cases, the logical assertions can view as a program rather than data to a program.
- Hencetheimplicationstatementsdefinethelegitimatereasoningpathsandautomatic assertions provide the starting points of those paths.
- Thesepathsdefinetheexecutionpathswhichissimilartothe'ifthenelse"intraditional programming.
- Sologicalassertionscanviewas aproceduralrepresentationof knowledge.

LogicProgramming–RepresentingKnowledgeUsing Rules

- Logicprogrammingisaprogrammingparadigminwhichlogicalassertionsviewedas programs.
- Theseareseverallogicprogrammingsystems,PROLOGisoneof them.
- ***APROLOGprogramconsistsofseverallogical assertionswhereeachisahorn clause i.e. a clausewith atmostonepositiveliteral.***
- Ex:P, PV Q, P→ Q
- Thefactsare represented on HornClausefortworeasons.
    1. Becauseof auniformrepresentation,asimpleandefficientinterpretercanwrite.
    2. Thelogicof HornClause decidable.

- Also,ThefirsttwodifferencesarethefactthatPROLOGprogramsareactuallysetsof Horn clause that have been transformed as follows:-
    1. IftheHornClausecontains nonegativeliteralthen leaveitasit is.
    2. Also,OtherwiserewritetheHornclausesasanimplication,combiningallofthe negative literals into the antecedent of the implications and the single positive literal into the consequent.
- Moreover,Thisprocedurecausesaclausewhichoriginallyconsistedofadisjunctionof literals (one of them was positive) to be transformed into a single implication whose antecedent is a conjunction universally quantified.
- But when we apply this transformation, any variables that occurred in negative literals andsonowoccurintheantecedentbecomeexistentiallyquantified,whilethevariablesin the consequent are still universally quantified.

ForexamplethePROLOGclauseP(x):–Q(x,y)isequaltologicalexpression∀x:∃y:Q(x, y) → P(x).

- ThedifferencebetweenthelogicandPROLOGrepresentationisthatthePROLOG interpretation has a fixed control strategy. And so, the assertions in the PROLOG program define a particular search path to answer any question.
- But,thelogicalassertionsdefineonlythesetofanswersbutnotabouthowtochoose among those answers if there is more than one.

Considerthefollowing example:

1. Logicalrepresentation

∀x:pet(x)□small(x)→apartmentpet(x)
∀x:cat(x)□dog(x)→pet(x)
∀x:poodle(x)→dog(x)□small(x) poodle
(fluffy)

2. Prologrepresentation

apartmentpet(x):pet(x),small(x) pet
(x): cat (x)
pet(x):dog(x)
dog(x): poodle (x)
small (x): poodle(x)
poodle (fluffy)

ForwardversusBackwardReasoning

ForwardversusBackwardReasoning
Asearchproceduremustfindapathbetweeninitialandgoalstates. There are two directions in which a search process could proceed. The two types of search are:

1. Forwardsearchwhichstartsfromthestartstate
2. Backwardsearchthatstartsfromthe goalstate

Theproductionsystemviewstheforwardandbackwardassymmetricprocesses. Consider a game of playing 8 puzzles. The rules defined are

Square1 emptyand square2 contains tilen.→

- *Also,Square2emptyandsquare1containsthetilen.*

Square 1 empty Square 4 contains tile n. →

- *Also,Square 4 emptyand Square1 contains tilen.*

Wecan solvetheproblem in 2 ways:

1. Reasonforwardfromtheinitial state
   - Step1.Beginbuildingatreeofmovesequencesbystartingwiththeinitialconfiguration at the root of the tree.
   - Step 2. Generate the next level of the tree by finding all rules ***whose left-hand side matches***againsttherootnode.Theright-handsideisusedtocreatenewconfigurations.
   - Step3.Generatethenextlevelbyconsideringthenodesinthepreviousleveland applying it to all rules whose left-hand side match.
2. Reasoningbackwardfromthegoalstates:
   - Step1.Beginbuildingatreeofmovesequencesbystartingwiththegoalnode configuration at the root of the tree.
   - Step2.Generatethenextlevelofthetreebyfindingallrules***whoseright-handside matches***againsttherootnode.Theleft-handsideusedtocreatenewconfigurations.
   - Step3.Generatethenextlevelbyconsideringthenodesinthepreviousleveland applying it to all rules whose right-hand side match.
   - So,Thesame rules canusein both cases.
   - Also,Inforwardingreasoning,theleft-handsidesoftherulesmatchedagainstthecurrent state and right sides used to generate the new state.
   - Moreover,Inbackwardreasoning,theright-handsidesoftherulesmatchedagainstthe current state and left sides are used to generate the new state.

Thereare fourfactorsinfluencingthetypeof reasoning. Theyare,

1. Aretheremorepossiblestartorgoalstate? Wemovefromsmallersetofsetstothe length.
2. Inwhatdirectionisthebranchingfactorgreater?Weproceedinthedirectionwiththe lower branching factor.
3. Willtheprogrambeaskedtojustifyitsreasoningprocesstoauser?If,sothenitis selected since it is very close to the way in which the user thinks.
4. Whatkindofeventisgoingtotriggeraproblem-solvingepisode? Ifitisthearrivalofa new factor, the forward reasoning makes sense. If it is a query to which a response is desired, backward reasoning is more natural.

Example1ofForwardversusBackward Reasoning

- It is easier to drive from an unfamiliar place from home, rather than from home to an unfamiliarplace.Also,Ifyouconsiderahomeasstartingplaceanunfamiliarplaceasa goal then we have to backtrack from unfamiliar place to home.

Example2ofForwardversusBackward Reasoning

- Consider a problem of symbolic integration. Moreover, The problem space is a set of formulas,whichcontainsintegralexpressions.HereSTARTisequaltothegivenformula with some integrals. GOAL is equivalent to the expression of the formula without any integral. Here we start from the formula with some integrals and proceed to an integral free expression rather than starting from an integral free expression.

Example3ofForwardversusBackward Reasoning

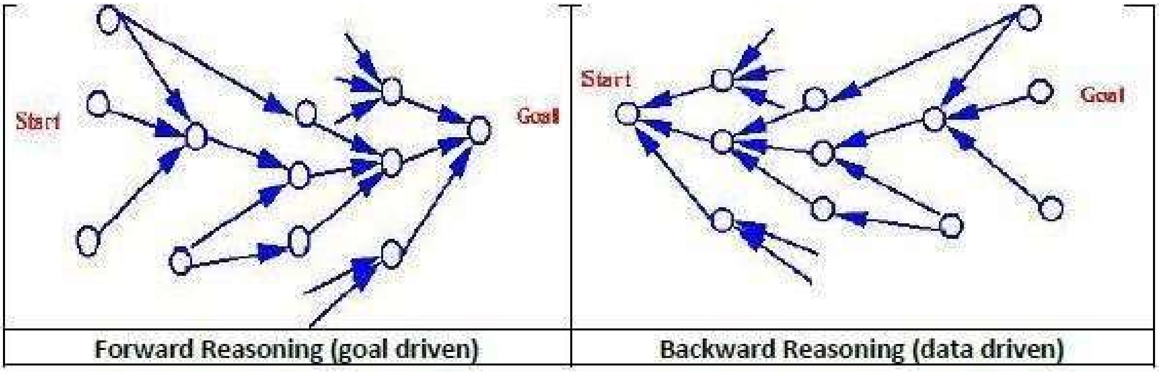
- The third factor is nothing but deciding whether the reasoning process can justify its reasoning.Ifitjustifiesthenitcanapply.Forexample,doctorsareusuallyunwillingto accept any advice from diagnostics process because it cannot explain its reasoning.

Example4ofForwardversusBackward Reasoning

- Prologisanexampleofbackwardchainingrulesystem.InPrologrulesrestrictedtoHorn clauses. This allows for rapid indexing because all the rules for deducing a given fact share the same rule head. Rules matched with unification procedure. Unification tries to find aset of bindings for variables to equate asub-goal with the head of some rule. Rules in the Prolog program matched in the order in which they appear.

CombiningForwardandBackwardReasoning

- Insteadofsearchingeitherforwardorbackward, youcansearchbothsimultaneously.
- Also,Thatis,startforwardfromastartingstateandbackwardfromagoalstate simultaneously until the paths meet.
- ThisstrategycalledBi-directionalsearch.Thefollowingfigureshowsthereasonfora Bidirectional search to be ineffective.


Start
Goal
Forward Reasoning (goal driven)
Start
Goal
Backward Reasoning (data driven)

ForwardversusBackwardReasoning

- Also,Thetwo searchesmaypass each otherresultingin more work.
- Basedontheformoftherulesonecandecidewhetherthesamerulescanapplytoboth forward and backward reasoning.
- Moreover, Ifleft-handsideandrightoftherulecontainpureassertionsthentherulecan reverse.
- Andso thesame rule canapplyto bothtypes of reasoning.
- Ifthe rightsideofthe rulecontains anarbitraryprocedurethen therulecannot reverse.
- So,Inthiscase,whilewritingtherulethecommitmenttoadirectionofreasoningmust make.

SymbolicReasoningUnderUncertainty

Symbolic Reasoning

- Thereasoningistheactofderivingaconclusionfromcertainpropertiesusingagiven methodology.
- Thereasoningisaprocessofthinking;reasoningislogicallyarguing; reasoningis drawing the inference.
- *Whenasystemisrequiredtodosomething,thatithasnotbeenexplicitlytoldhowtodo, it must reason. It must figure out what it needs to know from what it already knows.*

- ManytypesofReasoninghavebeenidentified andrecognized,butmanyquestions regarding their logical and computational properties still remain controversial.
- The popular methods of Reasoning include abduction, induction, model-based, explanationandconfirmation.Allofthemareintimatelyrelatedtoproblemsofbelief revision and theory development, knowledge absorption, discovery, and learning.

LogicalReasoning

- Logicisalanguageforreasoning. Itisacollectionofrulescalled Logicarguments,we use when doing logical reasoning.
- Thelogicreasoningistheprocessofdrawingconclusionsfrompremisesusingrulesof inference.
- Thestudyoflogicdividedintoformalandinformallogic.Theformallogicissometimes called symbolic logic.
- Symboliclogicisthestudyofsymbolicabstractions(construct)thatcapturetheformal features of logical inference by a formal system.
- Theformalsystemconsistsoftwocomponents,aformallanguageplusasetofinference rules.
- Theformal system hasaxioms. Axiomis asentencethat isalways truewithin thesystem.
- Sentencesderivedusingthesystem'saxiomsandrulesofderivationcalledtheorems.
- TheLogicalReasoningisofourconcerninAI.

Approaches to Reasoning

- Therearethreedifferentapproachestoreasoningunderuncertainties.
    1. Symbolicreasoning
    2. Statistical reasoning
    3. Fuzzylogicreasoning

Symbolic Reasoning

- Thebasisforintelligentmathematicalsoftwareistheintegrationofthe"powerof symbolic mathematical tools" with the suitable "proof technology".
- Mathematicalreasoningenjoysapropertycalledmonotonicity,thatsays,"Ifaconclusion follows from given premises A, B, C… then it also follows from any larger set of premises, as long as the original premises A, B, C.. included."
- Moreover, Humanreasoningisnot monotonic.
- Peoplearriveatconclusionsonlytentatively;basedonpartialorincompleteinformation, reserve the right to retract those conclusions while they learn new facts. Such reasoning non-monotonic, precisely because the set of accepted conclusions have become smaller when the set of premises expanded.

Formal Logic

Moreover,TheFormallogicisthestudyofinferencewithpurelyformalcontent,i.e.where content made explicit.

Examples–PropositionallogicandPredicate logic.

- Herethelogicalargumentsareasetofrulesformanipulatingsymbols.Therulesareof two types,
    1. Syntax rules:sayhow tobuildmeaningful expressions.
    2. Inference rules:sayhow toobtaintrueformulasfromothertrueformulas.
- Moreover,Logicalsoneedssemantics,whichsayshowtoassignmeaningtoexpressions.

Uncertainty in Reasoning

- TheworldisanuncertainplaceMoftentheKnowledgeisimperfectwhichcauses uncertainty.
- So,Thereforereasoningmust beable tooperateunder uncertainty.
- Also,AIsystemsmusthavetheabilitytoreasonunderconditionsofuncertainty.

Monotonic Reasoning

- Areasoningprocessthatmovesinonedirection only.
- Moreover,Thenumberoffactsin theknowledge baseisalways increasing.
- Theconclusionsderivedarevaliddeductionsandtheyremainso. A

monotonic logic cannot handle

1. Reasoningbydefault:becauseconsequencesmayderiveonlybecauseoflackofevidence to the contrary.
2. Abductivereasoning:becauseconsequencesonlydeducedas mostlikelyexplanations.
3. Beliefrevision:because newknowledgemaycontradictoldbeliefs.

IntroductiontoNonmonotonicReasoning

Non-monotonic Reasoning

Thedefiniteclauselogicis**monotonic**inthesensethatanythingthatcouldbeconcludedbefore a clause is added can still be concluded after it is added; adding knowledge does not reduce the set of propositions that can be derived.

Alogicis**non-monotonic**ifsomeconclusionscanbeinvalidatedbyaddingmoreknowledge. The logic of definite clauses with negation as failure is non-monotonic. Non-monotonic reasoning is useful for representing defaults. A **default** is a rule that can be used unless it overridden by an exception.

Forexample,tosaythat *b*isnormallytrueif *c*istrue,aknowledgebasedesignercanwritearule of the form

$b \leftarrow c \wedge \sim ab_a$.

whereab_aisanatomthatmeansabnormalwithrespecttosomeaspect*a*.Given*c*,theagentcan infer *b*unless it is told ab_a. Adding $ab_a$to the knowledge base can prevent the conclusion of *b*. Rulesthatimplyab_acan beusedtopreventthedefaultundertheconditionsofthebodyofthe rule.

Example5.27: Supposethepurchasingagentisinvestigatingpurchasingholidays.Aresortmay beadjacenttoabeachor awayfromabeach.This isnotsymmetric;iftheresortwasadjacentto a beach, the knowledge provider would specify this. Thus, it is reasonable to have the clause

away_from_beach ← ~ on_beach.

Thisclauseenablesanagenttoinferthataresortisawayfromthebeachiftheagentisnottoldit is adjacent to a beach.

A **cooperative system** tries to not mislead. If we are told the resort is on the beach, we would expectthatresortuserswouldhaveaccesstothebeach.Iftheyhaveaccesstoabeach,wewould expect them to be able to swim at the beach. Thus, we would expect the following defaults:

beach_access ←on_beach ∧ ~ abbeach_access.

swim_at_beach←beach_access∧~abswim_at_beach.

A cooperative system would tell us if a resort on the beach has no beach access or if there is no swimming.Wecouldalsospecifythat,ifthereisanenclosedbayandabigcity,thenthereisno swimming, by default:

abswim_at_beach←enclosed_bay∧big_city∧~abno_swimming_near_city.

WecouldsaythatBritishColumbiaisabnormalwith respecttoswimmingnear cities:

abno_swimming_near_city←in_BC∧~abBC_beaches.
Given onlythe preceding rules, an agent infers *away_from_beach*. If it is then told *on_beach*, it cannolongerinfer*away_from_beach*,butitcannowinfer*beach_access*and*swim_at_beach*.If it is also told *enclosed_bay* and *big_city*, it can no longer infer *swim_at_beach*. However, if it is then told *in_BC*, it can then infer *swim_at_beach*.
Byhavingdefaultsofwhatisnormal,ausercaninteractwiththesystembytellingitwhatis abnormal, which allows for economyin communication. The user does not have to state the obvious.
One way to think about non-monotonic reasoning is in terms of **arguments**. The rules can be usedascomponentsofarguments,inwhichthenegatedabnormalitygivesawaytoundermine arguments. Note that, in the language presented, only positive arguments exist that can be undermined.Inmoregeneraltheories,therecanbepositiveandnegativeargumentsthatattack each other.

Implementation Issues

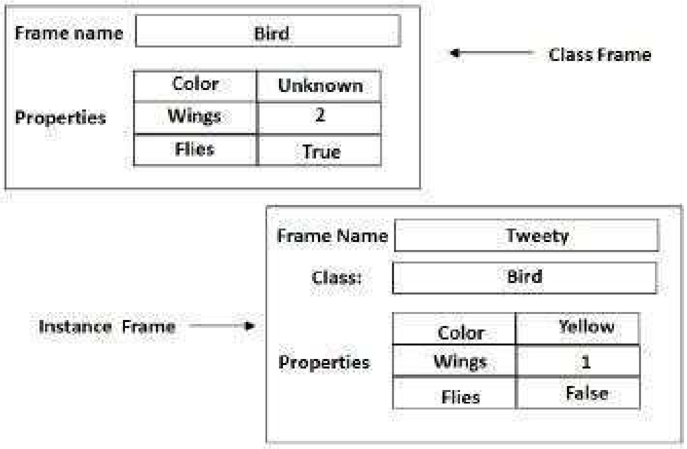
WeakSlotandFillerStructures

Evolution Frames

- Asseeninthepreviousexample,therearecertainproblemswhicharedifficulttosolve with Semantic Nets.
- Althoughthereisnocleardistinctionbetweenasemanticnetandframesystem,more structured the system is, more likely it is to be termed as a frame system.
- A frame is a collection of attributes (called slots) and associated values that describe someentitiesintheworld.Sometimesaframedescribesanentityinsomeabsolutesense;
- Sometimesit representstheentityfromaparticular pointof view only.
- Asingleframetakenaloneisrarelyuseful;webuildframesystemsoutofcollectionsof frames that connected to each other byvirtue of the fact that the value of an attribute of one frame may be another frame.

FramesasSetsand Instances

- Theset theoryis agoodbasis forunderstandingframe systems.
- Eachframerepresents eitheraclass(aset)oran instance(an elementofclass)
- Both***isa***and***instance***relationshaveinverseattributes,whichwecallsubclasses&all instances.
- As aclass represents a set, thereare2 kinds of attributes that can beassociated with it.
    1. Itsownattributes&
    2. Attributes that areto be inherited byeach element of theset.



FramesasSetsand Instances

- Sometimes,the differencebetween aset andan individual instancemaynot be clear.
- Example:TeamIndiaisaninstanceoftheclassofCricketTeamsandcanalsothinkofas the set of players.
- NowtheproblemisifwepresentTeamIndiaasasubclassofCricketteams,thenIndian players automatically become part of all the teams, which is not true.
- So,wecanmakeTeamIndiaasubclassof classcalledCricket Players.
- Todothis weneed todifferentiatebetweenregular classesand meta-classes.
- RegularClassesarethosewhoseelementsareindividualentitieswhereasMeta-classes are those special classes whose elements are themselves, classes.
- Themost basicmeta-class isthe class *CLASS*.
- Itrepresentsthesetofallclasses.
- Allclasses areinstancesof it, eitherdirectlyor throughoneof its subclasses.
- TheclassCLASS*CLASS*introducestheattributecardinality,whichistoinheritedbyallinstances of CLASS. Cardinality stands for the number.

OtherwaysofRelatingClasses toEach Other

- Wehavediscussed that aclass1can beasubset ofclass2.
- If Class2isameta-classthenClass1canbe aninstanceof Class2.
- Anotherwayisthe***mutually-disjoint-with***relationship,whichrelatesaclasstooneor more other classes that guaranteed to have no elements in common with it.
- Anotheroneis,***is-covered-by***whichrelatesaclasstoasetofsubclasses,theunionof which is equal to it.
- Ifaclassis-covered-byasetSofmutuallydisjointclasses,thenScalledapartitionofthe class.

SlotsasFull-FledgedObjects (Frames)

Tillnowwehaveusedattributesasslots,butnowwewillrepresentattributesexplicitlyand describe their properties.

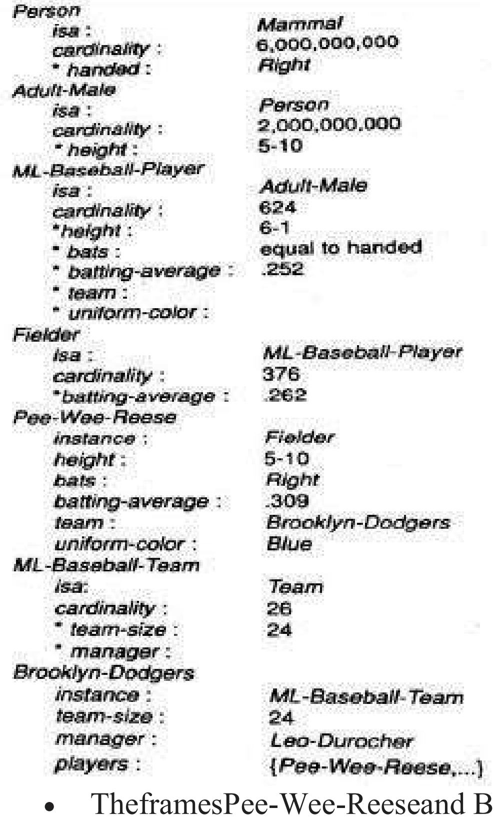
Someoftheproperties wewould liketo be abletorepresentand usein reasoning include,

- Theclass towhich the attributecan attach.
- Constraintsoneither thetypeorthevalue ofthe attribute.
- Adefaultvalue forthe attribute.Rules forinheritingvaluesfortheattribute.
- Tobeabletorepresenttheseattributesofattributes,weneedtodescribeattributes(slots) as frames.

- Theseframeswillorganizeintoan***isa***hierarchy,justasanyotherframes,andthat hierarchy can then used to support inheritance of values for attributes of slots.
- Nowlet us formalizewhat is aslot. A slot hereis a relation.
- Itmapsfromelementsofitsdomain(theclassesforwhichitmakessense)toelementsof its range (its possible values).
- Arelationis aset ofordered pairs.
- Thusit makessenseto saythat relationR1 is asubset ofanother relation R2.
- Inthat case,R1isaspecializationofR2.Sinceaslotisaset,thesetofallslots,which we will call SLOT, is a meta-class.
- Itsinstances areslots, whichmayhavesub-slots.

FrameExample

In this example, the frames Person, Adult-Male, ML-Baseball-Player (corresponding to major leaguebaseballplayers),Pitcher,andML-Baseball-Team(formajorleaguebaseballteam)areall classes.



```
Person
    isa :                      Mammal
    cardinality :              6,000,000,000
    * handed :                 Right
Adult-Male
    isa :                      Person
    cardinality :              2,000,000,000
    * height :                 5-10
ML-Baseball-Player
    isa :                      Adult-Male
    cardinality :              624
    *height :                  6-1
    * bats :                   equal to handed
    * batting-average :        .252
    * team :
    * uniform-color :
Fielder
    isa :                      ML-Baseball-Player
    cardinality :              376
    *batting-average :         .262
Pee-Wee-Reese
    instance :                 Fielder
    height :                   5-10
    bats :                     Right
    batting-average :          .309
    team :                     Brooklyn-Dodgers
    uniform-color :            Blue
ML-Baseball-Team
    isa:                       Team
    cardinality :              26
    * team-size :              24
    * manager :
Brooklyn-Dodgers
    instance :                 ML-Baseball-Team
    team-size :                24
    manager :                  Leo-Durocher
    players :                  {Pee-Wee-Reese,...}
```

- TheframesPee-Wee-Reeseand Brooklyn-Dodgersareinstances.
- The***isa***relationthatwehavebeenusingwithoutaprecisedefinitionis,infact,thesubset relation. The set of adult males is a subset of the set of people.
- Theset ofmajor leaguebaseball playerssubset of the setof adultmales, andso forth.
- Ourinstancerelationcorrespondstotherelationelement-ofPeeWeeReeseisanelement of the set of fielders.
- Thus he is also an element of all of the supersets of fielders, including major league baseballplayersandpeople.Thetransitivityof*isa*followsdirectlyfromthetransitivity of the subset relation.

- Boththeisaandinstancerelationshaveinverseattributes,whichwecallsubclassesand all instances.
- Becauseaclassrepresentsaset,therearetwokindsofattributesthatcanassociatewith it.
- Someattributesareaboutthesetitself,andsomeattributesaretoinheritedbyeach element of the set.
- Weindicate thedifferencebetween thesetwo byprefixingthe latter withanasterisk (*).
- Forexample,considertheclassML-Baseball-Player,wehaveshownonlytwoproperties of it as a set: It a subset of the set of adult males. And it has cardinality 624.
- Wehavelistedfivepropertiesthatallmajorleaguebaseballplayershave(height,bats, battingaverage,team,anduniform-color),andwehavespecifieddefaultvaluesforthe first three of them.
- Byprovidingbothkindsofslots,weallowbothclassestodefineasetofobjectsandto describe a prototypical object of the set.
- Framesareusefulforrepresentingobjectsthataretypicalofstereotypicalsituations.
- Thesituationlikethestructureofcomplexphysicalobjects,visualscenes,etc.
- Acommonsenseknowledgecanrepresentusingdefaultvaluesifnoothervalueexists. Commonsense is generally used in the absence of specific knowledge.

SemanticNets

- Inheritancepropertycanrepresent using**isa**and **instance**
- MonotonicInheritancecanperformsubstantiallymoreefficientlywithsuchstructures than with pure logic, and non-monotonic inheritance is also easily supported.
- ThereasonthatmakesInheritanceeasyisthattheknowledgeinslotandfillersystemsis structured as a set of entities and their attributes.

Thesestructuresturn outto beuseful as,

- Itindexesassertionsbytheentitiestheydescribe. Asaresult,retrievingthevalueforan attribute of an entity is fast.
- Moreover, Itmakeseasytodescribepropertiesofrelations.Todothisinapurelylogical system requires higher-order mechanisms.
- Itisaformofobject-orientedprogrammingandhastheadvantagesthatsuchsystems normally include modularity and ease of viewing by people.

Herewewould describetwo viewsofthis kind ofstructure – Semantic Nets & Frames.

SemanticNets

- Therearedifferentapproachestoknowledgerepresentationincludesemanticnet,frames, and script.
- Thesemanticnetdescribes bothobjectsand events.
- Inasemanticnet,informationrepresentedasasetofnodesconnectedtoeachotherbya set of labeled arcs, which represents relationships among the nodes.
- Itisadirectedgraphconsistingofverticeswhichrepresentconceptsandedgeswhich represent semantic relations between the concepts.
- Itis alsoknownasassociativenet dueto theassociation of onenodewith other.
- Themainideaisthatthemeaningoftheconceptcomesfromthewaysinwhichit connected to other concepts.
- Wecan useinheritancetoderiveadditional relations.

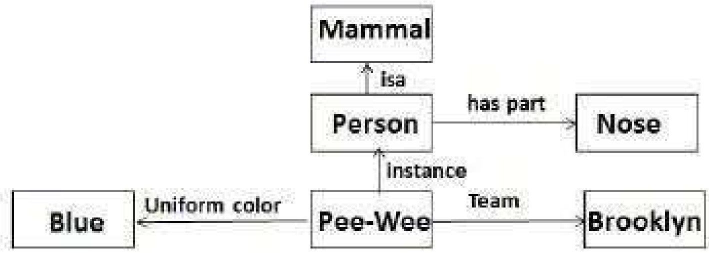


Figure:ASemanticNetwork

IntersectionSearchSemanticNets

- Wetrytofindrelationshipsamongobjectsbyspreadingactivationoutfromeachoftwo nodes. And seeing where the activation meets.
- Usingthiswecananswer thequestionslike, whatistherelationbetweenIndiaand Blue.
- Ittakesadvantageoftheentity-basedorganizationofknowledgethatslotandfiller representation provides.

RepresentingNon-binaryPredicates SemanticNets

- Simplebinarypredicateslikeisa(Person,Mammal)canrepresenteasilybysemanticnets but other non-binary predicates can also represent by using general-purpose predicates such as ***isa*** and ***instance***.
- Threeorevenmoreplacepredicatescanalsoconverttoabinaryformbycreatingone new object representing the entire predicate statement and then introducing binary predicates to describe a relationship to this new object.

ConceptualDependency

IntroductiontoStrongSlotandFillerStructures

- Themainproblemwithsemanticnetworksandframesisthattheylackformality;thereis no specific guideline on how to use the representations.
- Inframewhenthingschange,weneedtomodifyallframesthatarerelevant –thiscanbe time-consuming.
- Strong slot and filler structures typically represent links between objects according to morerigidrules,specificnotionsofwhattypesofobjectandrelationsbetweenthemare provided and represent knowledge about common situations.
- Moreover,Wehavetypesof strongslotand fillerstructures:
  1. Conceptual Dependency(CD)
  2. Scripts
  3. Cyc

ConceptualDependency(CD)

ConceptualDependencyoriginallydevelopedtorepresentknowledgeacquiredfromnatural language input.

Thegoals of this theory are:

- Tohelpin thedrawingof the inferencefrom sentences.
- Tobeindependent of thewords usedin theoriginal input.
- Thatistosay:Forany2(ormore)sentencesthatareidenticalinmeaningthereshouldbe only one representation of that meaning.

Moreover, Ithasusedbymanyprogramsthatportendtounderstand English(MARGIE, SAM, PAM).

ConceptualDependency(CD) provides:

- Astructureintowhich nodesrepresentinginformationcan beplaced.
- Also,A specificset of primitives.
- Agivenlevelofgranularity.

Sentencesarerepresentedasaseriesofdiagramsdepictingactionsusingbothabstractandreal physical situations.

- Theagentand theobjectsrepresented.
- Moreover,Theactionsarebuiltupfromasetofprimitiveactswhichcanmodifyby tense.

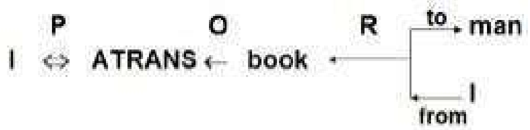
CDisbased onevents andactions. Everyevent (if applicable)has:

- anACTOR oan ACTION performedbythe Actor
- Also,anOBJECTthattheactionperformson
- ADIRECTIONinwhich thatactionisoriented

Thesearerepresentedasslotsandfillers. InEnglishsentences,manyoftheseattributesleft out.

ASimpleConceptualDependency Representation

Forthesentences,"Ihaveabook totheman"CDrepresentationisasfollows:



Wherethesymbolshavethefollowingmeaning.

- Arrowsindicatedirectionsofdependency.
- Moreover,Thedouble arrowindicates thetwo-waylinkbetween actor andaction.
- O— fortheobject caserelation
- R–fortherecipientcaserelation
- P– forpast tense
- D– destination

PrimitiveActsofConceptualDependency Theory

ATRANS

- Transferofanabstractrelationship(i.e.give)

PTRANS

- Transferofthephysicallocationofanobject(e.g.,go)

PROPEL

- Also,Applicationofphysicalforcetoanobject(e.g.push)

MOVE

- Moreover,Movementofabodypartbyitsowner(e.g.kick)

GRASP

- Graspingofanobjectbyanaction(e.g.throw)

INGEST

- Ingestingofanobjectbyananimal(e.g.eat)

EXPEL

- Expulsionofsomethingfromthebodyofananimal(e.g.cry)

MTRANS

- Transferofmentalinformation(e.g.tell)

MBUILD

- Buildingnewinformationoutofold(e.gdecide)

SPEAK

- Producingofsounds(e.g.say)

ATTEND

- Focusingof asenseorgantoward astimulus (e.g.listen)

Therearefourconceptualcategories.Theseare,

ACT

- Actions{oneoftheCD primitives}

PP

- Also,Objects{pictureproducers}

AA

- Modifiersofactions{action aiders}

PA

- ModifiersofPP's{pictureaiders}

Advantagesof ConceptualDependency

- Usingtheseprimitives involvesfewer inferencerules.
- So,Manyinferencerulesalreadyrepresentedin CD structure.
- Moreover,Theholes intheinitial structurehelp to focuson thepointsstillto established.

DisadvantagesofConceptualDependency

- Knowledgemust decomposeintofairlylow-levelprimitives.
- Impossibleordifficulttofindthecorrectsetof primitives.
- Also,A lot ofinferencemaystill require.
- Representationscanbecomplexeven forrelativelysimple actions.
- Consider:DavebetFrank fivepounds thatWales wouldwin theRugbyWorld Cup.
- Moreover,Complex representationsrequirealotofstorage.

Scripts

ScriptsStrongSlot

- Ascriptisastructurethatprescribesasetofcircumstanceswhichcouldbeexpectedto follow on from one another.
- Itissimilartoa thoughtsequenceora chainof situationswhich couldbe anticipated.
- Itcouldbeconsideredtoconsistofanumberofslotsorframesbutwithmorespecialized roles.

Scriptsarebeneficialbecause:

- Eventstendtooccur inknownrunsor patterns.
- Causalrelationshipsbetweeneventsexist.
- Entryconditionsexist whichallowan eventtotakeplace
- Prerequisitesexistforeventstakingplace.E.g.whenastudentprogressesthrougha degree scheme or when a purchaser buys a house.

ScriptComponents

Eachscriptcontains thefollowingmain components.

- EntryConditions: Must besatisfied beforeevents in the script can occur.
- Results:Conditions thatwill betrueafterevents inscript occur.
- Props:Slotsrepresentingobjectsinvolvedintheevents.
- Roles:Persons involvedin theevents.
- Track:theSpecificvariationonthemoregeneralpatterninthescript.Differenttracks may share many components of the same script but not all.

- Scenes:Thesequenceofeventsthatoccur.Eventsrepresentedinconceptualdependency form.

AdvantagesandDisadvantagesof Script

Advantages

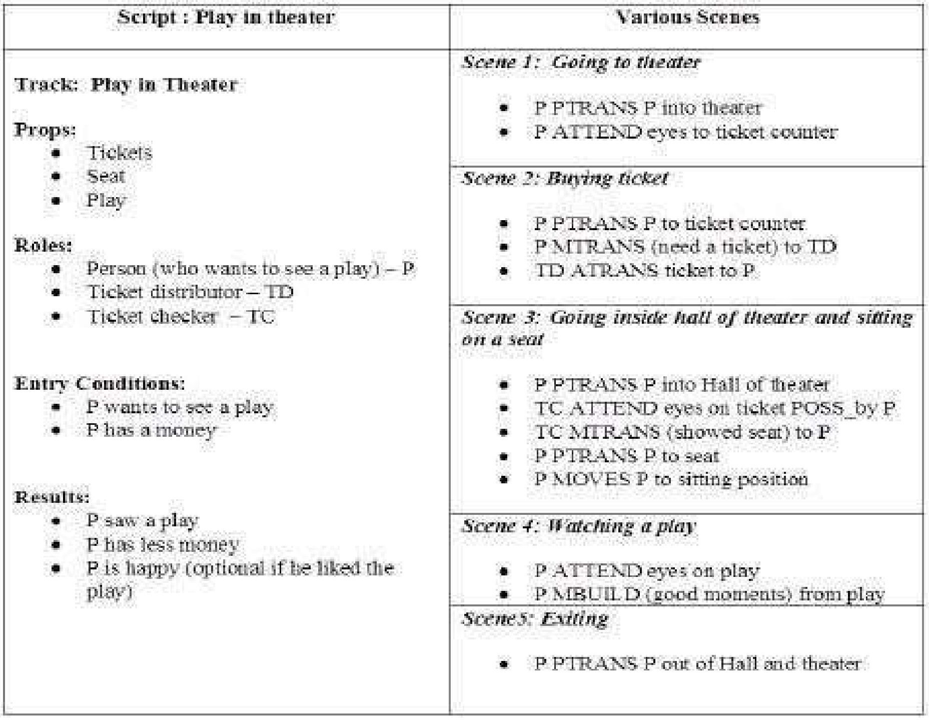
- Capable ofpredictingimplicitevents
- Singlecoherentinterpretationmaybebuildupfromacollectionofobservations.

Disadvantage

- Morespecific(inflexible)andlessgeneralthanframes.
- Notsuitabletorepresent allkinds of knowledge.

Todealwithinflexibility,smallermodulescalledmemoryorganizationpackets(MOP) can combine in a way that appropriates for the situation.

ScriptExample



Script : Play in theater	Various Scenes
Track: Play in Theater **Props:** • Tickets • Seat • Play **Roles:** • Person (who wants to see a play) – P • Ticket distributor – TD • Ticket checker – TC **Entry Conditions:** • P wants to see a play • P has a money **Results:** • P saw a play • P has less money • P is happy (optional if he liked the play)	***Scene 1: Going to theater*** • P PTRANS P into theater • P ATTEND eyes to ticket counter
	Scene 2: Buying ticket • P PTRANS P to ticket counter • P MTRANS (need a ticket) to TD • TD ATRANS ticket to P
	Scene 3: Going inside hall of theater and sitting on a seat • P PTRANS P into Hall of theater • TC ATTEND eyes on ticket POSS_by P • TC MTRANS (showed seat) to P • P PTRANS P to seat • P MOVES P to sitting position
	Scene 4: Watching a play • P ATTEND eyes on play • P MBUIL D (good moments) from play
	Scene5: Exiting • P PTRANS P out of Hall and theater

- Itmustactivatebasedonitssignificance.
- Ifthetopicimportant, thenthescriptshouldopen.
- If atopicjust mentioned, then apointerto thatscript could hold.
- Forexample,given“Johnenjoyedtheplayintheater”,ascript“PlayinTheater” suggested above invoke.
- Allimplicitquestionscananswercorrectly.

Here the significance of this script is high.

- DidJohn go to thetheater?
- Also, Didhe buythe ticket?
- Didhehavemoney?

Ifwehaveasentencelike“Johnwenttothetheatertopickhisdaughter”,theninvokingthis script will lead to many wrong answers.

- Heresignificanceofthescripttheateris less.

Gettingsignificancefromthestoryisnotstraightforward.However,someheuristicscan applyto get the value.

CYC

What is CYC?

- Anambitiousattempttoformaverylargeknowledgebaseaimedatcapturing commonsense reasoning.
- Initialgoalstocaptureknowledgefromahundredrandomlyselectedarticlesinthe Encyclopedia Britannica.
- Also,Both ImplicitandExplicitknowledge encoded.
- Moreover,Emphasisonstudyofunderlyinginformation(assumedbytheauthorsbutnot needed to tell to the readers.

Example:SupposewereadthatWellingtonlearnedofNapoleon’sdeath □Thenwe(humans) can conclude Napoleon never new that Wellington had died.

Howdo wedo this?

So,Werequirespecialimplicit knowledgeorcommonsensesuch as:

- We onlydieonce.
- You staydead.
- Moreover,Youcannotlearnanythingwhendead.
- Timecannot gobackward.

Whybuildlargeknowledgebases:

1. Brittleness
   - Specialisedknowledgebasesarebrittle.Hardtoencodenewsituationsandnon-graceful degradation in performance. Commonsense based knowledge bases should have a firmer foundation.
2. Formand Content
   - Moreover,KnowledgerepresentationmaynotbesuitableforAI.Commonsense strategies could point out where difficulties in content may affect the form.
3. SharedKnowledge
   - Also,Shouldallowgreatercommunicationamongsystemswithcommonbases and assumptions.

Howis CYCcoded?

- By hand.
- Special CYCLlanguage:
- LISP-like.
- Frame-based
- Multipleinheritances
- Slotsarefullyfledged objects.
- Generalizedinheritance—anylink notjust ***isa*** and ***instance***.

Module2

GamePlaying:

Game Playing

- CharlesBabbage,thenineteenth-centurycomputerarchitectthoughtaboutprogramming his analytical engine to play chess and later of building a machine to play tic-tac-toe.
- Therearetworeasonsthatgamesappearedtobe agood domain.
    1. Theyprovideastructuredtaskinwhichitisveryeasytomeasure success or failure.
    2. Theyareeasilysolvablebystraightforwardsearchfromthestartingstatetoa winning position.
- Thefirst istrueisforallgames bust thesecond isnot trueforall,except simplest games.
- Forexample,considerchess.
- Theaveragebranchingfactorisaround35.Inanaveragegame,eachplayermightmake 50.
- Soinordertoexaminethecompletegametree,wewouldhavetoexamine35^{100}
- Thusitisclearthatasimplesearchisnotabletoselectevenitsfirstmoveduringthe lifetime of its opponent.
- Itisclearthattoimprovetheeffectivenessofasearchbasedproblem-solvingprogram two things can do.
    1. Improvethe generateproceduresothat onlygoodmoves generated.
    2. Improvethetestproceduresothatthebestmove will recognizeandexplored first.
- Ifweuselegal-movegeneratorthenthetestprocedurewillhavetolookateachofthem because the test procedure must look at so many possibilities, it must be fast.
- Insteadofthelegal-movegenerator,wecanuseplausible-movegeneratorinwhichonly some small numbers of promising moves generated.
- Asthenumberoflawyersavailablemovesincreases,itbecomesincreasinglyimportant in applying heuristics to select only those moves that seem more promising.
- Theperformanceoftheoverallsystemcanimprovebyaddingheuristicknowledgeinto both the generator and the tester.
- Ingameplaying,agoalstateisoneinwhichwewinbutthegamelikechess.Itisnot possible. Even we have good plausible move generator.
- Thedepth of theresultingtreeorgraphand its branchingfactor is too great.
- Itispossibletosearchtreeonlytenortwentymovesdeeptheninordertochoosethebest move. The resulting board positions must compare to discover which is most advantageous.
- This is done using static evolution function, which uses whatever information it has to evaluateindividualboardpositionbyestimatinghowlikelytheyaretolead eventuallyto a win.
- Itsfunctionissimilartothatoftheheuristicfunctionh'intheA*algorithm:inthe absence of complete information, choose the most promising position.

MINIMAXSearchProcedure

- Theminimaxsearchisa depth-firstanddepthlimitedprocedure.
- Theideaistostartatthecurrentpositionandusetheplausible-movegeneratorto generate the set of possible successor positions.

- Nowwecanapplythestaticevolutionfunctiontothosepositionsandsimplychoosethe best one.
- Afterdoingso,wecanbackthatvalueuptothestartingpositiontorepresentour evolution of it.
- Hereweassumethatstaticevolutionfunctionreturnslargervaluestoindicategood situations for us.
- Soourgoalistomaximizethevalueofthestaticevaluationfunctionofthenextboard position.
- Theopponents'goalistominimizethevalue ofthestaticevaluationfunction.
- **Thealternationofmaximizingandminimizingatalternateplywhenevaluationsare to be pushed back up corresponds to the opposing strategies of the two players is called MINIMAX.**
- Itis therecursiveprocedurethat dependson twoprocedures
  - MOVEGEN(position,player)—Theplausible-movegenerator,whichreturnsa list of nodes representing the moves that can make by Player in Position.
  - STATIC(position,player)–staticevaluationfunction,whichreturnsanumber representing the goodness of Position from the standpoint of Player.
- Withanyrecursiveprogram,weneed todecidewhen recursiveprocedureshould stop.
- There arethevarietyoffactorsthat mayinfluencethe decision they are,
  - Hasoneside won?
  - Howmanyplies havewe alreadyexplored?Orhow much time is left?
  - Howstableis theconfiguration?
- WeuseDEEP-ENOUGHwhichassumedtoevaluateallofthesefactorsandtoreturn TRUE if the search should be stopped at the current level and FALSE otherwise.
- Ittakestwoparameters,position,anddepth,itwillignoreitspositionparameterand simply return TRUE if its depth parameter exceeds a constant cut off value.
- OneproblemthatarisesindefiningMINIMAXasarecursiveprocedureisthatitneedsto return not one but two results.
  - Thebacked-up valueof thepath it chooses.
  - The path itself. We return the entire path even though probably only the first element,representingthebestmovefromthecurrentposition,actuallyneeded.
- WeassumethatMINIMAXreturnsastructurecontainingbothresults andwehavetwo functions, VALUE and PATH that extract the separate components.
- Initially,Ittakesthreeparameters,aboardposition,thecurrentdepthofthesearch,and the player to move,
  - MINIMAX(current,0,player-one)Ifplayer–oneistomove
  - MINIMAX(current,0,player-two)Ifplayer–twoistomove

Addingalpha-beta cutoffs

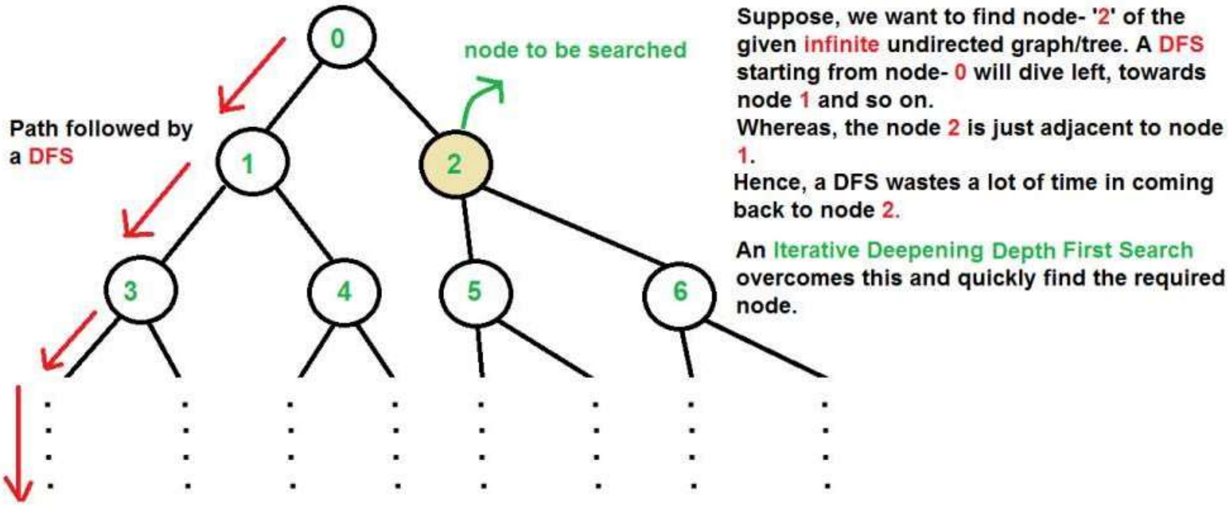
- Minimax procedureisa depth-firstprocess.One pathisexploredasfaras timeallows, thestaticevolutionfunctionisappliedtothegamepositionsatthelaststepofthepath.
- Theefficiencyofthedepth-firstsearchcanimprovebybranchandboundtechniquein which partial solutions that clearly worse than known solutions can abandon early.
- Itisnecessarytomodifythebranchandboundstrategytoincludetwobounds,onefor each of the players.
- Thismodifiedstrategycalledalpha-beta pruning.

- Itrequiresmaintainingoftwothresholdvalues,onerepresentingalowerboundonthata maximizing node may ultimately assign (we call this alpha).
- Andanotherrepresentinganupperboundonthevaluethataminimizingnodemayassign (this we call beta).
- Eachlevelmustreceivebothethevalues,onetouseandonetopassdowntothenextlevel to use.
- TheMINIMAXprocedureasitstandsdoesnotneedtotreatmaximizingandminimizing levels differently. Since it simply negates evaluation each time it changes levels.
- Insteadofreferringtoalphaandbeta,MINIMAXusestwovalues,USE-THRESHand PASSTHRESH.
- USE-THRESHusedtocomputecutoffs.PASS-THRESHpassedtonextlevelasits USETHRESH.
- USE-THRESHmustalsopasstothenextlevel,butitwillpassasPASS-THRESHsothat it can be passed to the third level down as USE-THRESH again, and so forth.
- Justas valueshad tonegate eachtime theypassedacross levels.
- Still,thereisnodifferencebetweenthecoderequiredatmaximizinglevelsandthat required at minimizing levels.
- PASS-THRESHshouldalwaysthemaximumofthevalueitinheritsfromaboveandthe best move found at its level.
- If PASS-THRESH updated the new value should propagate both down to lower levels. Andbackuptohigheronessothatitalwaysreflectsthebestmovefoundanywhereinthe tree.
- TheMINIMAX-A-Brequiresfivearguments,position,depth,player,Use-thresh,and passThresh.
- MINIMAX-A-B(current,0,player-one,maximumvaluestaticcancompute,minimum value static can compute).

IterativeDeepeningSearch(IDS)orIterativeDeepeningDepthFirstSearch(IDDFS)

Therearetwocommonwaystotraverseagraph, BFSandDFS.ConsideringaTree(orGraph) of huge height and width, both BFS and DFS are not very efficient due to following reasons.

1. **DFS** first traverses nodes going through one adjacent of root, then next adjacent. The problemwiththisapproachis,ifthereisanodeclosetoroot,butnotinfirstfewsubtrees explored byDFS, then DFS reaches that node verylate. Also, DFS maynot find shortest path to a node (in terms of number of edges).



Suppose, we want to find node- '2' of the given infinite undirected graph/tree. A DFS starting from node- 0 will dive left, towards node 1 and so on.
Whereas, the node 2 is just adjacent to node 1.
Hence, a DFS wastes a lot of time in coming back to node 2.

An Iterative Deepening Depth First Search overcomes this and quickly find the required node.

2. **BFS** goes level by level, but requires more space. The space required by DFS is O(d) wheredisdepthoftree, butspacerequiredbyBFSisO(n)wherenisnumberofnodesin tree (Why? Note that the last level of tree can have around n/2 nodes and second lastlevel n/4 nodes and in BFS we need to have every level one by one in queue).

IDDFScombinesdepth-firstsearch'sspace-efficiencyandbreadth-firstsearch'sfastsearch(for nodes closer to root).

HowdoesIDDFSwork?

IDDFScallsDFSfordifferentdepthsstartingfromaninitial value.Ineverycall,DFSis restricted from going beyond given depth. So basically we do DFS in a BFS fashion.

Algorithm:

```
//Returnstrueiftargetisreachablefrom
// srcwithin max_depth
boolIDDFS(src,target, max_depth)
   forlimitfrom0 to max_depth
     ifDLS(src,target,limit)==true
        return true
   returnfalse

boolDLS(src,target,limit)
   if(src==target)
      returntrue;

   // Ifreachedthemaximumdepth,
   // stop recursing.
   if (limit<=0)
      returnfalse;

   foreachadjacentiofsrc
```

```
    ifDLS(i,target, limit?1)
        return true

 returnfalse
```

An important thing to note is, we visit top level nodes multiple times. The last (or max depth) level is visited once, second last level is visited twice, and so on. It mayseem expensive, but it turnsouttobenotsocostly,sinceinatreemostofthenodesareinthebottomlevel.Soitdoes not matter much if the upper levels are visited multiple times.

Planning

BlocksWorldProblem

Inordertocomparethevarietyofmethodsofplanning,weshouldfinditusefultolookatallof them in a single domain that is complex enough that the need for each of the mechanisms is apparent yet simple enough that easy-to-follow examples can be found.

- Thereisa flat surfaceonwhich blockscanbeplaced.
- Thereareanumberof squareblocks, all thesamesize.
- Theycan bestacked oneupon the other.
- Thereis robot arm thatcan manipulatethe blocks.

Actionsof therobotarm

1. UNSTACK(A,B):PickupblockAfromitscurrentpositiononblock B.
2. STACK(A,B):PlaceblockAonblockB.
3. PICKUP(A):Pickup blockA fromthetableandhold it.
4. PUTDOWN(A): Put block A down on the table.

Noticethattherobotarmcanholdonlyoneblockatatime.

Predicates

- Inordertospecifyboththeconditionsunderwhichanoperationmaybeperformedand the results of performing it, we need the following predicates:
    1. ON(A,B):BlockAisonBlock B.
    2. ONTABLES(A):Block Aisonthe table.
    3. CLEAR(A):Thereis nothingonthe topof BlockA.
    4. HOLDING(A):ThearmisholdingBlockA.
    5. ARMEMPTY:Thearm is holding nothing.

Robotproblem-solvingsystems (STRIPS)

- Listofnewpredicates thattheoperator causestobecome trueisADDList
- Moreover, Listofoldpredicatesthattheoperator causestobecomefalseisDELETEList
- PRECONDITIONSlistcontainsthosepredicatesthatmustbetruefortheoperatortobe applied.

STRIPSstyleoperatorsforBLOCKsWorld

STACK(x,y)
P: CLEAR(y)^HOLDING(x)
D:CLEAR(y)^HOLDING(x)
A: ARMEMPTY^ON(x, y)
UNSTACK(x, y)
PICKUP(x)
P:CLEAR(x)^ONTABLE(x)^ARMEMPTY D:
ONTABLE(x) ^ ARMEMPTY

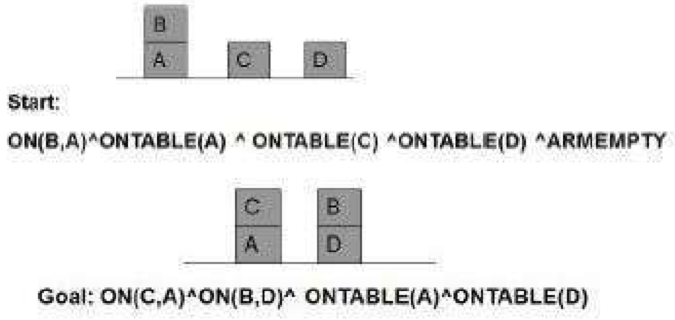
A:HOLDING(x)
PUTDOWN(x)

GoalStackPlanning

Tostartwith goalstackis simply:

- ON(C,A)^ON(B,D)^ONTABLE(A)^ONTABLE(D)

Thisproblemis separate intofour sub-problems,oneforeachcomponent of the goal.
Twoof thesub-problems ONTABLE(A) andONTABLE(D)arealreadytrueintheinitial state.



Start:
ON(B,A)^ONTABLE(A) ^ ONTABLE(C) ^ONTABLE(D) ^ARMEMPTY

Goal: ON(C,A)^ON(B,D)^ ONTABLE(A)^ONTABLE(D)

Alternative1:GoalStack:

- ON(C,A)
- ON(B,D)
- ON(C,A)^ON(B,D)^OTAD

Alternative2:Goalstack:

- ON(B,D)
- ON(C,A)
- ON(C,A)^ON(B,D)^OTAD

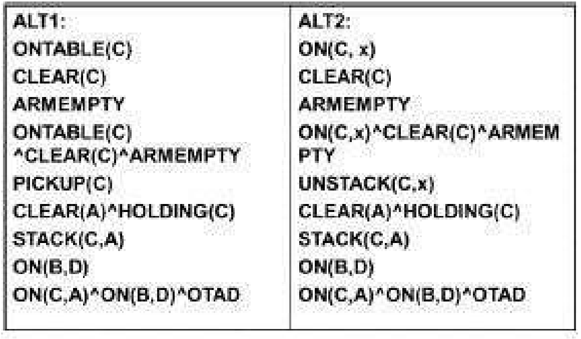
Exploring Operators

- Pursuingalternative1, wecheckforoperatorsthatcould cause ON(C, A)
- Outofthe 4operators, thereis onlyoneSTACK.So ityields:
    - STACK(C,A)
    - ON(B,D)
    - ON(C,A)^ON(B,D)^OTAD
- PreconditionsforSTACK(C, A)shouldbesatisfied, wemustestablish them as sub-goals:
    - CLEAR(A)
    - HOLDING(C)
    - CLEAR(A)^HOLDING(C)
    - STACK(C,A)oON(B,D)
    - ON(C,A)^ON(B,D)^OTAD
- HereweexploittheHeuristicthatifHOLDINGisoneoftheseveralgoalstobeachieved at once, it should be tackled last.

GoalstackPlanning

- Next,weseeifCLEAR(A)istrue. Itisnot.Theonlyoperatorthatcouldmakeittrueis UNSTACK(B, A). Also, This produces the goal stack:
    - ON(B,A)
    - CLEAR(B)
    - ON(B,A)^CLEAR(B)^ARMEMPTY
    - UNSTACK(B,A)
    - HOLDING(C)

  - CLEAR(A)^HOLDING(C)
  - STACK(C,A)
  - ON(B,D)
  - ON(C,A)^ON(B,D)^OTAD
- Weseethatwecanpoppredicatesonthestacktillwereach HOLDING(C)forwhichwe need to find a suitable operator.
- Moreover, The operators that might make HOLDING(C) true: PICKUP(C) and UNSTACK(C,x).Withoutlookingahead,sincewecannottellwhichoftheseoperators is appropriate. Also, we create two branches of the search tree corresponding to the following goal stacks:



ALT1:	ALT2:
ONTABLE(C)	ON(C, x)
CLEAR(C)	CLEAR(C)
ARMEMPTY	ARMEMPTY
ONTABLE(C) ^CLEAR(C)^ARMEMPTY	ON(C,x)^CLEAR(C)^ARMEM PTY
PICKUP(C)	UNSTACK(C,x)
CLEAR(A)^HOLDING(C)	CLEAR(A)^HOLDING(C)
STACK(C,A)	STACK(C,A)
ON(B,D)	ON(B,D)
ON(C,A)^ON(B,D)^OTAD	ON(C,A)^ON(B,D)^OTAD

Completeplan

1. UNSTACK(C,A)
2. PUTDOWN(C)
3. PICKUP(A)
4. STACK(A,B)
5. UNSTACK(A,B)
6. PUTDOWN(A)
7. PICKUP(B)
8. STACK(B, C)
9. PICKUP(A)
10. STACK(A,B)

PlanningComponents

- Methods which focus on ways of decomposing the original problem into appropriate subpartsandonwaysofrecordingandhandlinginteractionsamongthesubpartsasthey are detected during the problem-solving process are often called as planning.
- Planningreferstotheprocessofcomputingseveralstepsofaproblem-solvingprocedure before executing any of them.

Componentsofaplanning system

Choosethe best rule toapplynext, based on the best available heuristicinformation.

- Themostwidelyusedtechniqueforselectingappropriaterulestoapplyisfirsttoisolate a set of differences between desired goal state and then to identify those rules that are relevant to reduce those differences.
- Iftherearesseveralrules,avarietyofotherheuristicinformationcanbeexploitedto choose among them.

Applythechosenruletocomputethe new problemstate that arisesfrom its application.

- Insimplesystems,applyingrulesiseasy.Eachrulesimplyspecifiestheproblemstate that would result from its application.
- Incomplexsystems,wemustbeabletodealwithrulesthatspecifyonlyasmallpartof the complete problem state.
- Onewayistodescribe,foreachaction,eachofthechangesit makestothestate description.

Detectwhenasolutionhas found.

- Aplanningsystemhassucceededinfindingasolutiontoaproblemwhenithasfounda sequence of operators that transform the initial problem state into the goal state.
- Howwill it knowwhen this has done?
- Insimpleproblem-solvingsystems,thisquestioniseasilyansweredbyastraightforward match of the state descriptions.
- Oneoftherepresentativesystemsforplanningsystemsispredicatelogic.Supposethatas a part of our goal, we have the predicate P(x).
- ToseewhetherP(x)satisfiedinsomestate,weaskwhetherwecanproveP(x)giventhe assertions that describe that state and the axioms that define the world model.

Detectdeadendssothattheycanabandonandthesystem’seffortdirectedinmorefruitful directions.

- As a planning system is searching for a sequence of operators to solve a particular problem,itmustbeabletodetectwhenitisexploringapaththatcanneverleadtoa solution.
- Thesamereasoningmechanismsthatcanusetodetectasolutioncanoftenusefor detecting a dead end.
- Ifthesearchprocessisreasoningforwardfromtheinitialstate.Itcanpruneanypaththat leads to a state from which the goal state cannot reach.
- If search process reasoning backward from the goal state, it can also terminate a path eitherbecauseitissurethattheinitialstatecannotreachorbecauselittleprogressmade.

Detectwhenanalmostcorrectsolutionhasfoundandemployspecialtechniquestomakeit totally correct.

- Thekindsoftechniquesdiscussedareoftenusefulinsolvingnearlydecomposable problems.
- One good way of solving such problems is to assume that they are completely decomposable,proceedtosolvethesub-problemsseparately.Andthencheckthatwhen the sub-solutions combined. They do in fact give a solution to the original problem.

GoalStackPlanning

- Methods which focus on ways of decomposing the original problem into appropriate subpartsandonwaysofrecording.Andhandlinginteractionsamongthesubpartsasthey are detected during the problem-solving process are often called as planning.
- Planningreferstotheprocessofcomputingseveralstepsofaproblem-solvingprocedure before executing any of them.
-

GoalStackPlanningMethod

- Inthismethod,theproblemsolvermakesuseofasinglestackthatcontainsbothgoals and operators. That have proposed to satisfy those goals.

- Theproblemsolveralsoreliesonadatabasethatdescribesthecurrentsituationandaset of operators described as PRECONDITION, ADD and DELETE lists.
- Thegoalstackplanningmethodattacksproblemsinvolvingconjoinedgoalsbysolving the goals one at a time, in order.
- Aplangeneratedbythismethodcontainsasequenceofoperatorsforattainingthefirst goal, followed by a complete sequence for the second goal etc.
- Ateachsucceedingstepoftheproblem-solvingprocess,thetopgoalonthestackwill pursue.
- Whenasequenceofoperatorsthatsatisfiesit,found,thatsequenceappliedtothestate description, yielding new description.
- Next,thegoalthatthenatthetopofthestackexplored.Andanattemptmadetosatisfyit, starting from the situation that produced as a result of satisfying the first goal.
- Thisprocesscontinuesuntilthegoalstackisempty.
- Thenasonelastcheck,theoriginalgoalcomparedtothefinalstatederivedfromthe application of the chosen operators.
- Ifanycomponentsofthegoalnotsatisfiedinthatstate.Thenthoseunsolvedpartsofthe goal reinserted onto the stack and the process resumed.

NonlinearPlanningusingConstraintPosting

- Difficultproblemscausegoalinteractions.
- Theoperatorsusedtosolveonesub-problemmayinterferewiththesolutiontoaprevious sub-problem.
- Mostproblemsrequireanintertwinedplaninwhichmultiplesub-problemsworkedon simultaneously.
- Suchaplaniscallednonlinearplanbecauseitisnotcomposedofalinearsequenceof complete sub-plans.

ConstraintPosting

- The idea of constraint posting is to build up a plan by incrementally hypothesizing operators,partialorderingsbetweenoperators,andbindingofvariableswithinoperators.
- Atanygiventimeintheproblem-solvingprocess, wemayhaveasetofusefuloperators butperhapsnoclearideaofhowthoseoperatorsshouldorderwithrespecttoeachother.
- Asolutionisapartiallyordered,partiallyinstantiatedsetofoperatorstogeneratean actual plan. And we convert the partial order into any number of total orders.

ConstraintPostingversusStateSpace search

StateSpace Search

- Moves in thespace: Modifyworld stateviaoperator
- Modeloftime: Depthof nodeinsearch space
- PlanstoredinSeriesofstatetransitions

Constraint Posting Search

- Movesinthespace:Addoperators,OderOperators,BindvariablesOrOtherwise constrain plan
- Modelof Time:Partiallyorderedset of operators
- PlanstoredinSingle node

Algorithm:NonlinearPlanning (TWEAK)

1. InitializeStobetheset ofpropositionsinthegoalstate.
2. RemovesomeunachievedpropositionPfrom S.

3. Moreover,AchievePbyusingstepaddition,promotion,DEclobbering,simple establishment or separation.
4. Reviewallthestepsintheplan,includinganynewstepsintroducedbystepaddition,to see if any of their preconditions unachieved. Add to S the new set of unachieved preconditions.
5. Also, IfSisempty,completetheplanbyconvertingthepartialorderofstepsintoatotal order, instantiate any variables as necessary.
6. Otherwise,gotostep2.

HierarchicalPlanning

- Inorder tosolvehard problems,aproblemsolvermayhavetogenerate longplans.
- Itisimportanttobeabletoeliminatesomeofthedetailsoftheproblemuntilasolution that addresses the main issues is found.
- Thenanattemptcanmaketofillintheappropriatedetails.
- Earlyattemptstodothisinvolvedtheuseofmacrooperators,inwhichlargeroperators were built from smaller ones.
- Inthisapproach,nodetailseliminatedfromactualdescriptionsoftheoperators.

ABSTRIPS

AbetterapproachdevelopedinABSTRIPSsystemswhichactuallyplannedinahierarchyof abstraction spaces, in each of which preconditions at a lower level of abstraction ignored.
ABSTRIPSapproachisas follows:

- Firstsolvetheproblemcompletely,consideringonlypreconditionswhosecriticality value is the highest possible.
- Thesevalues reflect theexpected difficultyof satisfyingthe precondition.
- Todothis,doexactlywhatSTRIPSdid,butsimplyignorethepreconditionsoflower than peak criticality.
- Oncethisdone,usetheconstructedplanastheoutlineofacompleteplanandconsider preconditions at the next-lowest criticality level.
- Augmentthe plan with operatorsthat satisfythosepreconditions.
- Becausethisapproachexploresentireplansatonelevelofdetailbeforeitlooksatthe lower-level details of any one of them, it has called length-first approach.

Theassignmentofappropriatecriticalityvalueiscrucialtothesuccessofthishierarchical planning method.
Thosepreconditions that no operatorcan satisfyareclearlythemost critical.
Example, solving a problem of moving the robot, for applying an operator, PUSH-THROUGH DOOR,thepreconditionthatthereexistadoorbigenoughfortherobottogetthroughisofhigh criticality since there is nothing we can do about it if it is not true.

OtherPlanningTechniques

ReactiveSystems

- Theideaofreactivesystemsistoavoidplanningaltogether,andinstead,usethe observable situation as a clue to which one can simply react.
- Areactivesystemmusthaveaccesstoaknowledgebaseofsomesortthatdescribeswhat actions should be taken under what circumstances.
- Areactivesystemisverydifferentfromtheotherkindsofplanningsystemswehave discussed. Because it chooses actions one at a time.
- Itdoesnot anticipateandselectanentireactionsequencebeforeitdoesthe firstthing.
- TheexampleisaThermostat.Thejobofthethermostatistokeepthetemperature constant inside a room.
- Reactivesystems arecapableofsurprisinglycomplexbehaviors.
- Themainadvantagereactivesystemshaveovertraditionalplannersisthattheyoperate robustly in domains that are difficult to model completely and accurately.
- Reactivesystemsdispensewithmodelingaltogetherandbasetheiractionsdirectlyon their perception of the world.
- Anotheradvantageofreactivesystemsisthattheyareextremelyresponsivesincethey avoid the combinatorial explosion involved in deliberative planning.
- Thismakes themattractiveforreal-time taskssuch asdriving and walking.

Other PlanningTechniques

Triangletables

- Providesawayofrecordingthegoalsthat eachoperatorexpectedtosatisfyaswellasthe goals that must be true for it to execute correctly.

Meta-planning

- Atechniqueforreasoningnotjustabouttheproblemsolvedbutalsoabouttheplanning process itself.

Macro-operators

- Allowaplannertobuildnewoperatorsthatrepresentcommonlyusedsequencesof operators.

Case-basedplanning:

- Re-usesoldplansto makenew ones.

UNDERSTANDING

Understanding is the simplest procedure of all human beings. Understanding means ability to determine some new knowledge from a given knowledge. For each action of a problem, the mapping of some new actions is verynecessary. Mapping the knowledge means transferring the knowledge from one representation to another representation. For example, if you will say “I need to go to New Delhi” for which you will book the tickets. The system will have “understood” if it finds the first available plane to New Delhi. But if you will saythe same thing to you friends, who knows that your familylives in “New Delhi”, he/she will have “understood” if he/she realizes that there may be a problem or occasion in your family. For people, understanding applies to inputs from all the senses. Computer understanding has so far been applied primarilyto images, speech and typed languages. It is important to keep in mind that the successorfailureofan“understanding”problemcanrarelybemeasuredinanabsolutesensebut must instead be measured with respect to a particular task to be performed. There are some factors that contribute to the difficulty of an understanding problem.

(a) Ifthetargetrepresentationisverycomplexforwhichyoucannotmapfromtheoriginal representation.

(b) Therearedifferenttypesofmappingfactorsmayariselikeone-to-one,one-to-manyand many to many.
(c) Somenoise ordisturbingfactors arealso there.
(d) Thelevel of interaction ofthe sourcecomponents maybecomplexone.
(e) Theproblemsolvermightbe unknownaboutsomemorecomplexproblems.
(f) Theintermediaryactionsmayalso beunavailable.

Consider an example of an English sentence which is being used for communication with a keywordbased data retrieval system. Suppose Iwant to know all about the temples in India. So I wouldneedtobetranslatedintoarepresentationsuchasTheabovesentenceisasimplesentence for which the corresponding representation maybe easyto implement. But what for the complex queries?

Considerthefollowingquery.
"Ramtold Sitahewould not eatapplewith her.Hehastogo to theoffice".

This type of complex queries can be modeled with the conceptual dependency representation which is more complex than that of simple representation. Constructing these queries is very difficult since more informationare to be extracted. Extracting more information will require somemoreknowledge.Alsothetypeofmappingprocessisnotquiteeasytotheproblemsolver.
Understanding is the process of mapping an input from its original form to a more useful one. Thesimplest kind ofmappingis "one-toone".
In one-to-one mapping each different problems would lead to only one solution. But there are veryfewinputswhichareone-to-one.Othermappingsarequitedifficulttoimplement.Many-to- one mappings are frequent is that free variation is often allowed, either because of the physical limitations of that produces the inputs or because such variation simply makes the task of generating the inputs.
Manytoonemappingrequirethattheunderstandingsystemknowaboutallthewaysthatatarget representation can be expressed in the source language. One-to-many mapping requires a great deal of domain knowledge in order to make the correct choice among the available target representation.
Themappingprocessissimplestifeachcomponentcanbemappedwithoutconcern fortheother components of the statement. If the number of interactions increases, then the complexityof the problem will increase. In many understanding situations the input to which meaning should be assigned is not always the input that is presented to the under stander.
Because of the complex environment in which understanding usually occurs, other things often interferewiththebasicinputbeforeitreachestheunderstander.Hencetheunderstandingwillbe more complex if there will be some sort of noise on the inputs.

NaturalLanguageProcessing

IntroductiontoNaturalLanguageProcessing

- Languagemeantforcommunicatingwiththe world.
- Also,Bystudyinglanguage,wecan come to understand moreabout theworld.
- Ifwecansucceedatbuildingcomputationalmodeoflanguage,wewillhaveapowerful tool for communicating with the world.
- Also,Welookathowwecanexploitknowledgeabouttheworld,incombinationwith linguistic facts, to build computational natural language systems.

NaturalLanguageProcessing(NLP)problemcandivideintotwotasks:

1. Processingwrittentext,usinglexical,syntacticandsemanticknowledgeofthelanguage as well as the required real-world information.
2. Processing spoken language, using all the information needed above plus additional knowledgeaboutphonologyaswellasenoughaddedinformationtohandlethefurther ambiguities that arise in speech.

StepsinNaturalLanguage Processing

MorphologicalAnalysis

- Individualwordsanalyzedintotheircomponentsandnon-wordtokenssuchas punctuation separated from the words.

SyntacticAnalysis

- Linearsequencesofwordstransformedintostructuresthatshowhowthewordsrelateto each other.
- Moreover,Somewordsequencesmayrejectiftheyviolatethelanguage’srulefor how words may combine.

SemanticAnalysis

- Thestructurescreatedbythesyntacticanalyzer assignedmeanings.
- Also,A mappingmadebetween thesyntacticstructuresand objects inthe task domain.
- Moreover,Structuresforwhichnosuchmappingpossiblemayreject.

Discourse integration

- Themeaningofanindividualsentencemaydependonthesentencesthatprecedeit.And also, may influence the meanings of the sentences that follow it.

PragmaticAnalysis

- Moreover,Thestructurerepresentingwhatsaidreinterpretedtodeterminewhatwas actually meant.

Summary

- Resultsofeachofthemainprocesses combinetoformanaturallanguagesystem.
- Alloftheprocessesareimportantinacomplete naturallanguageunderstanding system.
- Notall programs arewritten withexactlythese components.
- Sometimestwoor moreofthem collapsed.
- Doingthatusuallyresultsinasystemthatiseasiertobuildforrestrictedsubsetsof English but one that is harder to extend to wider coverage.

StepsNaturalLanguageProcessing

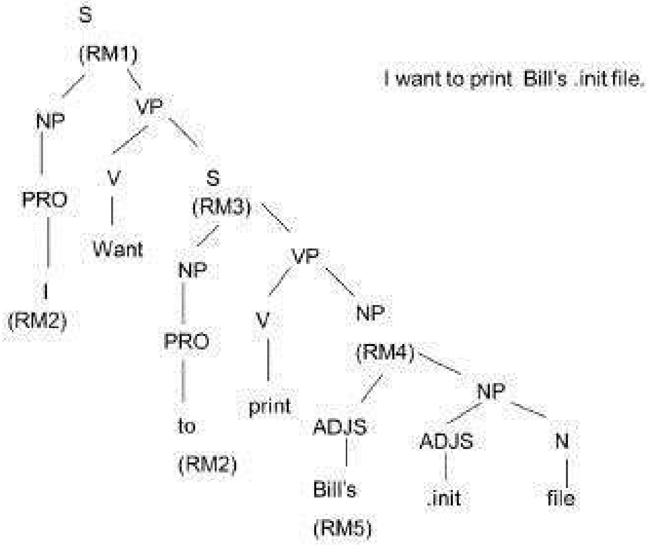
MorphologicalAnalysis

- SupposewehaveanEnglishinterfacetoanoperatingsystemandthefollowingsentence typed: I want to print Bill’s .init file.
- Themorphologicalanalysismustdothefollowingthings:

- Pullapart theword “Bill’s”intoproper noun“Bill”and thepossessivesuffix“’s”
- Recognizethesequence“.init”asafileextensionthatisfunctioningasanadjectiveinthe sentence.
- Thisprocesswillusuallyassignsyntacticcategories toallthe wordsinthe sentence.

SyntacticAnalysis

- Asyntacticanalysismustexploittheresultsofthemorphologicalanalysistobuilda structural description of the sentence.
- Thegoalofthisprocess,calledparsing,istoconverttheflatlistofwordsthatformthe sentence into a structure that defines the units that represented by that flat list.
- The important thing here is that a flat sentence has been converted into a hierarchical structure.Andthatthestructurecorrespondstomeaningunitswhenasemanticanalysis performed.
- Referencemarkers (set ofentities) shownintheparenthesisin theparse tree.
- Eachonecorresponds tosomeentitythat hasmentioned in the sentence.
- Thesereferencemarkersareusefullatersincetheyprovideaplaceinwhichto accumulate information about the entities as we get it.



SemanticAnalysis

- Thesemanticanalysismust dotwoimportant things:
    1. Itmustmapindividualwordsintoappropriateobjectsintheknowledgebaseor database.
    2. Itmustcreatethecorrectstructurestocorrespondtothewaythemeaningsofthe individual words combine with each other.

DiscourseIntegration

- Specifically,wedonotknowwhomthepronoun “I” ortheproper noun“Bill”refersto.
- To pin down these references requires an appeal to a model of the current discourse context,fromwhichwecanlearnthatthecurrentuserisUSER068andthattheonly person named “Bill” about whom we could be talking is USER073.
- OncethecorrectreferentforBillknown,wecanalsodetermineexactlywhichfile referred to.

PragmaticAnalysis

- Thefinal step towardeffectiveunderstandingis todecide what todo as aresult.

- Onepossible thingto dotorecord what wassaidasafactanddonewith it.
- Forsomesentences,awhoseintendedeffectisclearlydeclarative,thatistheprecisely correct thing to do.
- Butforothersentences, includingthis one,the intendedeffect is different.
- Wecandiscoverthisintendedeffectbyapplyingasetofrulesthatcharacterize cooperative dialogues.
- Thefinalstepinpragmaticprocessingtotranslate,fromtheknowledge-based representation to a command to be executed by the system.

SyntacticProcessing

- Syntactic Processing is the step in which a flat input sentence converted into a hierarchicalstructurethatcorrespondstotheunitsofmeaninginthesentence.This process called parsing.
- Itplays animportantrole innaturallanguageunderstandingsystemsfortworeasons:
    1. Semanticprocessingmustoperateonsentenceconstituents.Ifthereisnosyntactic parsing step, then the semantics system must decide on its own constituents. If parsing is done, on the other hand, it constrains the number of constituents that semantics can consider.
    2. Syntacticparsingiscomputationallylessexpensivethanissemanticprocessing. Thus it can play a significant role in reducing overall system complexity.
- Althoughitisoftenpossibletoextractthemeaningofasentencewithoutusing grammatical facts, it is not always possible to do so.
- Almostall thesystems thatareactuallyused have two main components:
    1. Adeclarativerepresentation,calledagrammar,ofthesyntacticfactsaboutthe language.
    2. Aprocedure,calledparserthatcomparesthegrammaragainstinputsentencesto produce parsed structures.

Grammarsand Parsers

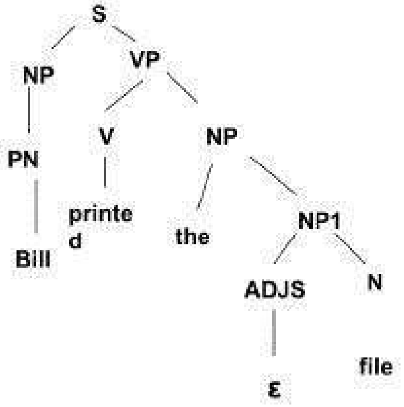
- Themost commonwaytorepresent grammars isasetof production rules.
- Thefirstrulecanreadas“AsentencecomposedofanounphrasefollowedbyVerb Phrase”; the Vertical bar is OR; ε represents the empty string.
- Symbolsthatfurther expandedbyrulescallednon-terminal symbols.
- Symbolsthatcorresponddirectlytostringsthatmustfoundinaninputsentencecalled terminal symbols.
- Grammarformalismsuchasthisoneunderliesmanylinguistictheories,whichinturn provide the basis for many natural language understanding systems.
- Purecontext-free grammarsarenoteffective fordescribingnatural languages.
- NLPshavelessincommonwithcomputerlanguageprocessingsystemssuchas compilers.
- Parsingprocesstakestherulesofthegrammarandcomparesthemagainsttheinput sentence.
- ThesimpleststructuretobuildisaParseTree,whichsimplyrecordstherulesandhow they matched.
- Everynodeoftheparsetreecorrespondseitherto aninputwordortoanon-terminalin our grammar.
- Eachlevelintheparsetreecorrespondstotheapplicationofone grammarrule.

ExampleforSyntacticProcessing–AugmentedTransition Network

Syntactic Processing is the step in which a flat input sentence is converted into a hierarchical structurethatcorrespondstotheunitsofmeaninginthesentence.Thisprocesscalledparsing. It plays an important role in natural language understanding systems for two reasons:

1. Semantic processing must operate on sentence constituents. If there is no syntactic parsingstep,thenthesemanticssystemmustdecideonitsownconstituents.Ifparsingis done, on the other hand, it constrains the number of constituents that semantics can consider.
2. Syntacticparsingiscomputationallylessexpensivethanissemanticprocessing.Thusit can play a significant role in reducing overall system complexity.

Example:A Parsetreeforasentence:BillPrintedthefile



Thegrammarspecifiestwothingsabouta language:

1. Its weak generative capacity, by which we mean the set of sentences that contained withinthelanguage.Thissetmadeupofpreciselythosesentencesthatcancompletely match by a series of rules in the grammar.
2. Itsstronggenerativecapacity,bywhichwemeanthestructuretoassigntoeach grammatical sentence of the language.

AugmentedTransitionNetwork (ATN)

- An augmented transition network is a top-down parsing procedure that allows various kindsofknowledgetoincorporatedintotheparsingsystemsoitcanoperateefficiently.
- ATNsbuild on the ideaof usingfinite statemachines (Markovmodel) to parse sentences.
- Insteadofbuildinganautomatonforaparticularsentence,acollectionoftransition graphs built.
- Agrammaticallycorrect sentenceparsed byreachingafinal state in anystategraph.
- Transitionsbetweenthesegraphssimplysubroutinecallsfromonestatetoanyinitial state on any graph in the network.
- Asentencedeterminedtobegrammaticallycorrectifafinalstatereachedbythelast word in the sentence.
- TheATNissimilartoafinitestatemachineinwhichtheclassoflabelsthatcanattach to the arcs that define the transition between states has augmented.

Arcs maylabelwith:

- Specificwordssuchas “in’.
- Wordcategoriessuchasnoun.

- Proceduresthatbuildstructuresthatwillformpartofthefinalparse.
- Proceduresthatperformarbitrarytestsoncurrentinputandsentencecomponentsthat have identified.

SemanticAnalysis

- Thestructurescreatedbythesyntacticanalyzer assignedmeanings.
- Amappingmadebetweenthesyntacticstructuresandobjects inthetask domain.
- Structuresforwhichnosuch mappingispossiblemayrejected.
- Thesemanticanalysismust dotwoimportant things:
  - Itmustmapindividualwordsintoappropriateobjectsintheknowledgebaseor database.
  - Itmustcreatethecorrectstructurestocorrespondtothewaythemeaningsofthe individual words combine with each other. Semantic Analysis AI
- Producing asyntacticparseof asentenceis onlythefirst step toward understandingit.
- Wemust producearepresentation ofthemeaningof the sentence.
- Becauseunderstandingisamappingprocess,wemustfirstdefinethelanguageinto which we are trying to map.
- Thereisnosingledefinitivelanguageinwhich allsentencemeaningcan describe.
- Thechoiceofatargetlanguageforanyparticularnaturallanguageunderstanding program must depend on what is to do with the meanings once theyconstructed.
- Choiceof thetarget languagein Semantic Analysis AI
  - There are two broad families of target languages that used in NL systems, dependingontherolethatthenaturallanguagesystemplayinginalargersystem:
  - When natural language considered as a phenomenon on its own, as for example when one builds a program whose goal is to read the text and then answer questionsaboutit.Atargetlanguagecandesignspecificallytosupportlanguage processing.
  - Whennaturallanguageusedasaninterfacelanguagetoanotherprogram(suchas a db querysystem or an expert system), then the target language must legal input to that other program. Thus the design of the target language driven by the backend program.

Discourse andPragmaticProcessing

Tounderstandasinglesentence,itisnecessarytoconsiderthediscourseandpragmaticcontext in which the sentence was uttered.
Thereareanumberofimportantrelationshipsthatmayholdbetweenphrasesandpartsoftheir discourse contexts, including:

1. Identicalentities.Considerthetext:
   - Billhad ared balloon. oJohn wanted it.
   - Theword“it”shouldidentifyasreferringtotheredballoon.Thesetypesof references called anaphora.
2. Partsofentities.Consider thetext:
   - Sueopened the bookshe just bought.
   - Thetitlepagewas torn.
   - Thephrase“titlepage”shouldberecognizedaspartofthebookthatwasjust bought.

3. Partsofactions.Considerthetext:
    - Johnwenton abusinesstrip toNew York.
    - Heleft onanearlymorning flight.
    - Takingaflight shouldrecognizeas partof goingonatrip.
4. Entitiesinvolvedinactions. Considerthetext:
    - Myhousewasbroken intolastweek.
    - Moreover, Theytook theTVand the stereo.
    - Thepronoun“they”shouldrecognizeasreferringtotheburglarswhobrokeinto the house.
5. Elementsofsets. Considerthetext:
    - Thedecals wehavein stock arestars, themoon, item and a flag.
    - I’lltaketwomoons.
    - Moonsmeanmoon decals.
6. Namesof individuals:
    - Devwenttothemovies.
7. Causalchains
    - Therewas a bigsnow stormyesterday.
    - So,Theschoolsclosed today.
8. Planningsequences:
    - Sallywantedanewcar
    - Shedecidedtogetajob.
9. Implicitpresuppositions:
    - Did Joefail CS101?

Themajorfocusis on usingfollowingkinds ofknowledge:

- Thecurrent focus of the dialogue.
- Also,Amodelofeachparticipant’scurrent beliefs.
- Moreover,Thegoal-drivencharacterofdialogue.
- Therulesofconversationshared byall participants.

StatisticalNaturalLanguageProcessing

Formerly, many language-processing tasks typically involved the direct hand coding of rules,whichisnotingeneralrobusttonatural-languagevariation.Themachine-learning paradigmcallsinsteadforusingstatisticalinferencetoautomaticallylearnsuchrulesthroughthe analysis of large ***corpora***of typical real-world examples (a *corpus* (plural, "corpora") is a set of documents, possibly with human or computer annotations).

Many different classes of machine learning algorithms have been applied to natural-language processingtasks.Thesealgorithmstakeasinputalargesetof"features"thataregeneratedfrom theinputdata.Someoftheearliest-usedalgorithms,suchasdecisiontrees,producedsystemsof hard if-then rules similar to the systems of hand-written rules that were then common. Increasingly,however,researchhasfocusedonstatisticalmodels,whichmake soft, probabilisticdecisions based on attaching real-valued weights to each input feature. Such models havethe advantagethat theycan express therelative certaintyof manydifferent possible answersratherthanonlyone,producingmorereliableresultswhensuchamodelisincludedasa component of a larger system.

Systemsbasedonmachine-learningalgorithmshavemanyadvantagesoverhand-producedrules:

- Thelearningproceduresusedduringmachinelearningautomaticallyfocusonthemost common cases, whereas when writingrules byhand it is often not at all obvious where the effort should be directed.
- Automatic learning procedures can make use of statistical inference algorithms to produce models that are robust to unfamiliar input (e.g. containing words or structures that have not been seen before) and to erroneous input (e.g. with misspelled words or wordsaccidentallyomitted).Generally,handlingsuchinputgracefullywithhand-written rules—or more generally, creating systems of hand-written rules that make soft decisions—is extremely difficult, error-prone and time-consuming.
- Systems based on automatically learning the rules can be made more accurate simply by supplying more input data. However, systems based on hand-written rules can only be made more accurate by increasing the complexity of the rules, which is a much more difficult task. In particular, there is a limit to the complexity of systems based on hand-crafted rules, beyond which the systems become more and more unmanageable. However, creating more data to input to machine-learning systems simply requires a correspondingincreaseinthenumberofman-hoursworked,generallywithoutsignificant increases in the complexity of the annotation process.

Spell Checking

Spell checking is one of the applications of natural language processing that impacts billions of usersdaily.AgoodintroductiontospellcheckingcanbefoundonPeterNorvig'swebpage.The article introduces a simple 21-line spell checker implementation in Python combining simple language and error models to predict the word a user intended to type. **The language model estimates how likely a given word `c` is in the language for which the spell checker is designed, this can be written as `P(C)`. The error model estimates the probability `P(w|c)` of typing the misspelled version `w` conditionally to the intention of typing the correctly spelled word `c`.**The spell checker then returns word `c` corresponding to the highest value of `P(w|c)P(c)`amongallpossiblewordsinthe language.

Module3

LEARNING

Learningistheimprovementofperformancewithexperienceover time.
LearningelementistheportionofalearningAIsystemthatdecideshowto modifythe performance element and implements those modifications.
We all learn new knowledge through different methods, depending on the type of material to be learned,theamountofrelevantknowledgewealreadypossess,andtheenvironmentinwhichthe learning takes place. There are five methods of learning . They are,

1. Memorization(rotelearning)
2. Directinstruction(bybeing told)
3. Analogy
4. Induction

5. Deduction
Learning by memorizations is the simplest from of le4arning. It requires the least amount of inferenceandisaccomplishedbysimplycopyingtheknowledgeinthesameformthatitwillbe used directly into the knowledge base.
Example:-Memorizingmultiplicationtables,formulate,etc.
Direct instruction is a complex form of learning. This type of learning requires more inference than role learning since the knowledge must be transformed into an operational form before learning when a teacher presents a number of facts directly to us in a well organized manner.
Analogical learning is the process of learning a new concept or solution through the use of similar known concepts or solutions. We use this type of learning when solving problems on an exam where previously learned examples serve as a guide or when make frequent use of analogicallearning.This formoflearningrequiresstillmoreinferringthan eitheroftheprevious forms.Sincedifficulttransformationsmustbemadebetweentheknownandunknownsituations.
Learning by induction is also one that is used frequently by humans . it is a powerful form of learning like analogical learning which also require s more inferring than the first two methods. This learning re quires the use of inductive inference, a form of invalid but useful inference. We use inductive learning ofinstances of examples of the concept. For example we learn theconcepts of color or sweet taste after experiencing the sensations associated with several examples of colored objects or sweet foods.
Deductive learning is accomplished through a sequence of deductive inference steps using knownfacts.Fromtheknownfacts,newfactsorrelationshipsarelogicallyderived.Deductive learning usually requires more inference than the other methods.
Review Questions:-
1. whatisperception?
2. Howdoweovercome thePerceptual Problems?
3. Explainindetailtheconstraintsatisfactionwaltz algorithm?
4. Whatislearning?
5. WhatisLearningelement?
6. Listandexplainthemethodsoflearning?
Types of learning:- Classification or taxonomyof learningtypes serves as a guide in studyingor comparingadifferencesamongthem.Onecandeveloplearningtaxonomiesbasedonthetypeof knowledge representation used (predicate calculus , rules, frames), the type of knowledge learned (concepts, game playing, problem solving), or by the area of application(medical diagnosis , scheduling , prediction and so on).
The classification is intuitively more appealing and is one which has become popular among machinelearningresearchers.itisindependentof theknowledgedomainandtherepresentation schemeisused.Itisbasedonthetypeofinferencestrategyemployedorthemethodsusedinthe learning process. The five different learning methods under this taxonomy are:
Memorization (rote learning)
Directinstruction(bybeingtold)
Analogy
Induction
Deduction
Learning by memorization is the simplest form of learning. It requires the least5 amount of inferenceandisaccomplishedbysimplycopyingtheknowledgeinthesameformthatitwillbe

useddirectlyintotheknowledgebase.Weusethistypeoflearningwhenwememorize multiplication tables ,
forexample.
Aslightlymorecomplexformoflearningisbydirectinstruction.Thistypeoflearningrequires more understanding and inference than role learning since the knowledge must be transformed into an operational form before being integrated into the knowledge base. We use this type of learning when a teacher presents a number of facts directly to us in a well organized manner.
The third type listed, analogical learning, is the process of learning an ew concept or solution through the use of similar known concepts or solutions. We use this type of learning when solving problems on an examination where previously learned examples serve as a guide or when we learn to drive a truck using our knowledge of car driving. We make frewuence use of analogicallearning.Thisformoflearningrequiresstillmoreinferringthaneitheroftheprevious forms,sincedifficulttransformationsmustbemadebetweentheknownandunknownsituations. This is a kind of application of knowledge in a new situation.
Thefourthtypeoflearningisalsoonethatisusedfrequencybyhumans.It isapowerfulformof learning which, like analogical learning, also requires more inferring than the first two methods. This form of learning requires the use of inductive inference, a form of invalid but useful inference. We use inductive learning when wed formulate a general concept after seeing a number of instance or examples of the concept. For example, we learn the concepts of color sweet taste after experiencing the sensation associated with several examples of colored objects or sweet foods.
The final type of acquisition is deductive learning. It is accomplished through a sequence of deductive inference steps using known facts. From the known facts, new facts or relationships arelogicallyderived.Deductivelearningusuallyrequiresmoreinferencethantheothermethods. The inference method used is, of course , a deductive type, which is a valid from of inference.
In addition to the above classification, we will sometimes refer to learning methods as wither methods or knowledge-rich methods. Weak methods are general purposemethods in which little or no initial knowledge is available. These methods are more mechanical than the classical AI knowledge–richmethods.Theyoftenrelyonaformofheuristicssearchinthelearningprocess.

Rote Learning

Rotelearningisthebasic learningactivity.**Rotelearning** isamemorizationtechnique based onrepetition. Itisalso calledmemorizationbecausethe knowledge, withoutanymodificationis, simplycopiedintotheknowledgebase.Ascomputedvaluesarestored,thistechniquecansavea significant amount of time.

Rote learning technique can also be used in complex learning systems provided sophisticated techniquesareemployedtousethestoredvaluesfasterandthereisageneralizationtokeepthe number of stored information down to a manageable level. Checkers-playing program, for ex

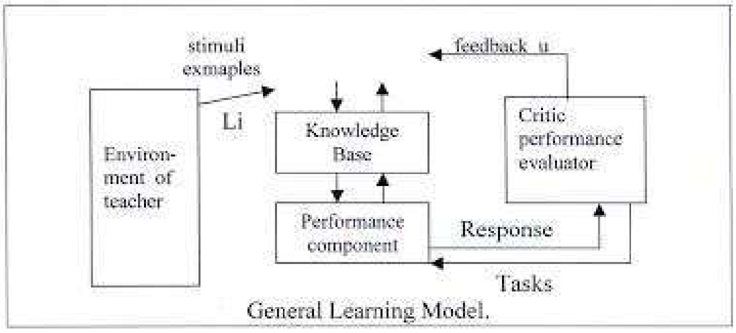
The idea is that one will be able to quickly recall the meaning of the material the more one repeats it. Some of the alternatives to rote learning include meaningful learning, associative learning,andactivelearning.ample,usesthistechniquetolearntheboardpositionsitevaluates in its look-ahead search.

LearningByTaking Advice.

Thisisasimpleformoflearning.Supposeaprogrammerwritesasetofinstructionstoinstruct the computer what to do, the programmer is a teacher and the computer is a student. Once learned (i.e. programmed), the system will be in a position to do new things.

The advice may come from many sources: human experts, internet to name a few. This type of learningrequiresmoreinferencethanrotelearning.Theknowledgemustbetransformedintoan operational form before stored in the knowledge base. Moreover the reliabilityof the source of knowledge should be considered.
The system should ensure that the new knowledge is conflicting with the existing knowledge. FOO(FirstOperationalOperationaliser),forexample,isalearningsystemwhichisusedtolearn the game of Hearts. It converts the advice which is in the form of principles, problems, and methods into effective executable (LISP) procedures (or knowledge). Now this knowledge is ready to use.

GeneralLearningModel.
General Learning Model: - AS noted earlier, learning can be accomplished using a number of different methods, such as by memorization facts, by being told, or by studying examples like problemsolution.Learningrequiresthatnewknowledgestructuresbecreatedfromsomeformof input stimulus. This new knowledge must then be assimilated into a knowledge base and be tested in some way for its utility. Testing means that the knowledge should be used in performance of some task from which meaningful feedback can be obtained, where the feedback provides some measure of the accuracy and usefulness of the newly acquired knowledge.
GeneralLearningModel



General Learning Model.

general learning model is depicted in figure 4.1 where the environment has been included as a part of the overall learner system. The environment may be regarded as either a form of nature which produces random stimuli or as a more organized training source such as a teacher which provides carefully selected training examples for the learner component. The actual form of environment used will depend on the particular learning paradigm. In any case, some representation language must be assumed for communication between the environment and the learner.Thelanguagemaybethesamerepresentationschemeasthatusedintheknowledgebase (such as a form of predicate calculus). When they are hosen to be the same, we say the single representation trick is being used. This usuallyresults in a simpler implementation since it is not necessary to transform between two or more different representations.

Forsomesystemstheenvironmentmaybeauserworkingatakeyboard.Othersystemswilluse program modules to simulate a particular environment. In even more realistic cases the system will have real physical sensors which interface with some world environment.

Inputstothelearnercomponentmaybephysicalstimuliofsometypeordescriptive,symbolic training examples. The information conveyed to the learner component is used to create and modify knowledge structures in the knowledge base. This same knowledge is used by the performance component to carryout some tasks, such as solving a problem playinga game, or classifying instances of some concept.

given a task, the performance component produces a response describing its action in performingthetask.Thecriticmodulethenevaluatesthisresponserelativetoanoptimal response.

Feedback,indicatingwhetherornottheperformancewasacceptable,isthensentbythecritic module to the learner component for its subsequent use in modifying the structures in the knowledge base. If proper learning was accomplished, the system's performance will have improved with the changes made to the knowledge base.

The cycle described above may be repeated a number of times until the performance of the systemhasreachedsomeacceptablelevel,untilaknownlearninggoalhasbeenreached,oruntil changes ceases to occur in the knowledge base after some chosen number of training examples have been observed.

Thereareseveralimportantfactorswhichinfluenceasystem'sabilitytolearninadditiontothe formofrepresentationused.Theyincludethetypesoftrainingprovided,theformandextentof any initial background knowledge , the type of feedback provided, and the learning algorithms used.

Thetypeoftrainingusedinasystemcanhaveastrongeffectonperformance,muchthesameas it does for humans. Training may consist of randomly selected instance or examples that have been carefullyselected and ordered for presentation. The instances maybe positive examples of some concept or task a being learned, they may be negative, or they may be mixture of both positive and negative. The instances may be well focused using only relevant information, or they may contain a variety of facts and details including irrelevant data.

There are Many forms of learning can be characterized as a search through a space of possible hypothesesorsolutions.Tomakelearningmoreefficient.Itisnecessarytoconstrainthissearch process or reduce the search space. One method of achieving this is through the use of background knowledge which can be used to constrain the search space or exercise control operations which limit the search process.

Feedback is essential to the learner component since otherwise it would never know if the knowledge structures in the knowledge base were improving or if they were adequate for the performanceofthegiventasks.Thefeedbackmaybeasimpleyesornotypeofevaluation, orit may contain more useful information describing why a particular action was good or bad. Also , thefeedbackmaybecompletelyreliable,providinganaccurateassessmentoftheperformanceor it maycontain noise, that is thefeedback mayactuallybe incorrect someof the time. Intuitively, the feedback must be accurate more than 50% of the time; otherwise the system carries useful information, the learner should also to build up a useful corpus of knowledge quickly. On the other hand, if the feedback is noisyor unreliable, the learning process may be very slow and the resultant knowledge incorrect.

LearningNeuralNetwork

Perceptron

- Theperceptronaninventionof(1962)Rosenblattwasoneoftheearliestneuralnetwork models.
- Also,Itmodelsaneuronbytakingaweightedsumofitsinputsandsendingtheoutput1 if the sum is greater than some adjustable threshold value (otherwise it sends 0).

$g(x) = \sum_{k=0}^{n} WkXk$	$\text{Output}(x) = \begin{cases} 1 & \text{if } g(x) > 0 \\ 0 & \text{if } g(x) > 0 \end{cases}$

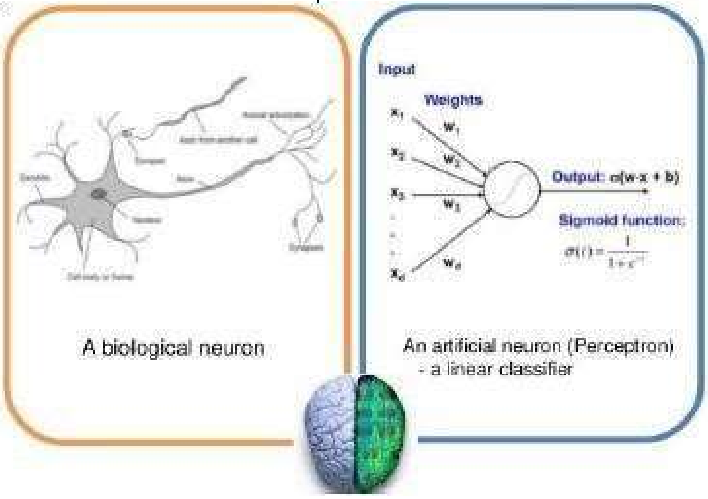


Figure:Aneuron&aPerceptron

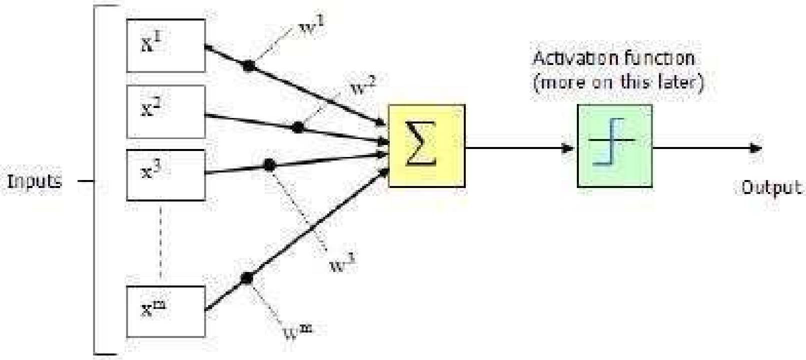


Figure:Perceptronwithadjustable threshold

- Incaseofzero withtwo inputsg(x) =w0 +w1x1 +w2x2 =0

- $x2=-(w1/w2)x1-(w0/w2)$→equation for aline
- thelocation ofthe lineis determined bytheweightw0w1 and w2
- ifan inputvector lies ononesideofthe line,theperceptronwill output 1
- ifit lies ontheotherside, theperception will output 0
- Moreover,Decisionsurface:alinethatcorrectlyseparatesthetraininginstances corresponds to a perfectly function perceptron.

PerceptronLearningAlgorithm

Given: A classification problem with n input feature (x_1, x_2, …., x_n) and two output classes.

ComputeAsetofweights(w_0,w_1,w_2,….,w_n)thatwillcauseaperceptrontofirewheneverthe input falls into the first output class.

1. Createaperceptronwithn+1input andn+1 weight,wherethex0is alwaysset to 1.
2. Initializetheweights(w_0,w_1,…., w_n)torandomreal values.
3. Iteratethroughthetrainingset,collectingallexamples *misclassified*bythecurrentsetof weights.
4. Ifallexamplesareclassifiedcorrectly,outputtheweightsandquit.
5. Otherwise,computethevectorsumSofthemisclassifiedinputvectorswhereeachvector has the form (x_0, x_1, …, Xn). In creating the sum, add to S a vector x if xis an input for which the perceptron incorrectly fails to fire, but – xif xis an input for which the perceptron incorrectly fires. Multiply sum by a scale factor η.
6. Moreover,Modifytheweights(w_0,w_1,…,w_n)byaddingtheelementsofthevectorSto them.
7. Goto step 3.

- The perceptron learning algorithm is a search algorithm. It begins with a random initial stateandfindsasolutionstate.Thesearchspaceissimplyallpossibleassignmentsofreal values to the weights of the perception, and the search strategy is gradient descent.
- Theperceptronlearningruleisguaranteedtoconvergetoasolutioninafinitenumberof steps, so long as a solution exists.
- Moreover, This brings us to an important question. What problems can a perceptron solve? Recallthatasingle-neuronperceptronisabletodividetheinputspaceintotwo regions.
- Also,Theperceptioncanbeusedtoclassifyinputvectorsthatcanbeseparatedbya linear boundary. We call such vectors linearly separable.
- Unfortunately,manyproblemsarenotlinearlyseparable.TheclassicexampleistheXOR gate. It was the inability of the basic perceptron to solve such simple problems that arenot linearly separable or non-linear.

Genetic Learning

Supervised Learning

Supervisedlearningisthemachinelearningtaskofinferringafunctionfromlabeledtraining data. Moreover,Thetrainingdata consist ofaset of training examples.

Insupervisedlearning,eachexampleapairconsistingofaninputobject(typicallyavector)and the desired output value (also called the supervisory signal).

Trainingset

Atrainingsetasetofdatausedinvariousareasofinformationsciencetodiscoverpotentially predictive relationships.

Trainingsetsusedinartificialintelligence,machinelearning,geneticprogramming,intelligent systems, and statistics.

In allthesefields,atrainingsethasmuchthesameroleandoftenusedinconjunctionwithatest set.

Testingset

A**testset**isasetofdatausedinvariousareasofinformationsciencetoassessthestrengthand utility of a predictive relationship.

Moreover,Testsetsareusedinartificialintelligence,machinelearning,geneticprogramming, and statistics. In all these fields, a test set has much the same role.

Accuracyofclassifier:Supervisedlearning

In the fields of science, engineering, industry, and statistics. The accuracy of a measurement systemisthedegreeofclosenessofmeasurementsofaquantitytothatquantity'sactual(true) value.

Sensitivityanalysis:Supervisedlearning

Similarly,LocalSensitivityascorrelationcoefficientsandpartialderivativescanonlyuse,ifthe correlation between input and output is linear.

Regression:Supervisedlearning

Instatistics,**regressionanalysis**isastatisticalprocessforestimatingtherelationshipsamong variables. Moreover,Itincludesmanytechniquesformodelingandanalyzingseveralvariables.Whenthe focus on therelationship between adependent variable and oneor moreindependent variables. More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when anyone of the independent variables varied. Moreover,While the other independent variables held fixed.

Expert systems:

Expertsystem=knowledge+problem-solvingmethods .... Aknowledgebasethat captures the domain-specific knowledge and an inference engine that consists of algorithms for manipulatingtheknowledgerepresentedintheknowledgebasetosolveaproblempresentedto the system.

Expertsystems(ES)areoneoftheprominentresearchdomainsofAI.Itisintroducedbythe researchers at Stanford University, Computer Science Department.

WhatareExpertSystems?

Theexpertsystemsarethecomputerapplicationsdevelopedtosolvecomplexproblemsina particular domain, at the level of extra-ordinary human intelligence and expertise.

CharacteristicsofExpertSystems

- Highperformance
- Understandable
- Reliable
- Highlyresponsive

CapabilitiesofExpert Systems

Theexpertsystemsarecapableof–

- Advising
- Instructingandassistinghumanindecisionmaking
- Demonstrating
- Derivinga solution
- Diagnosing
- Explaining
- Interpretinginput
- Predictingresults
- Justifyingtheconclusion
- Suggestingalternativeoptionstoaproblem

They are incapable of –

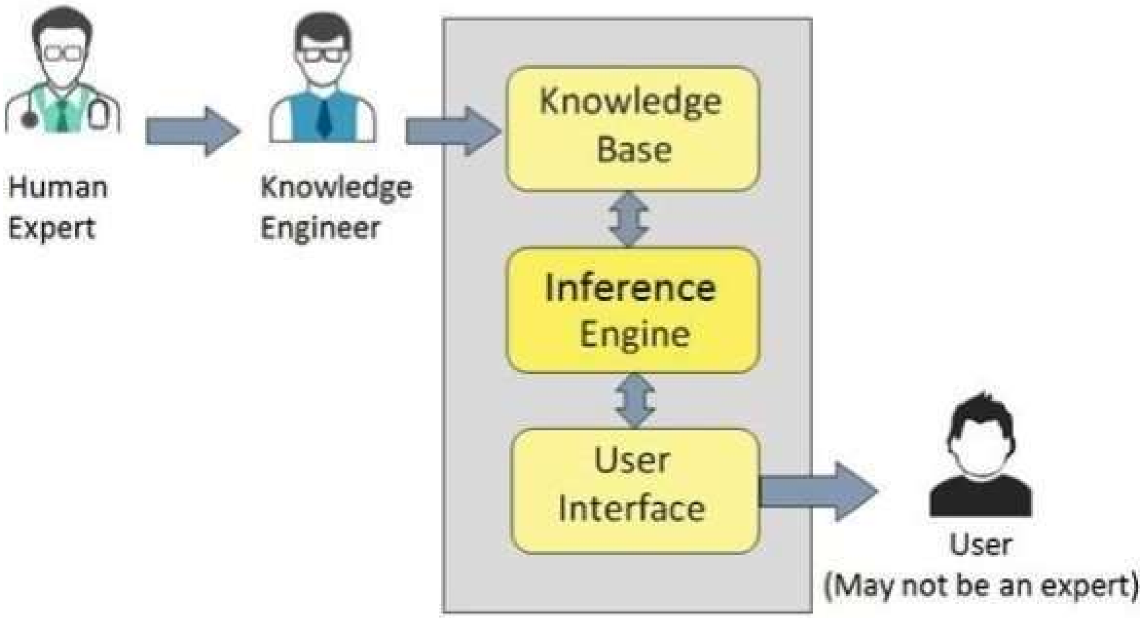
- Substitutinghumandecisionmakers
- Possessinghuman capabilities
- Producingaccurateoutputforinadequateknowledgebase
- Refiningtheirownknowledge

Components of Expert Systems

ThecomponentsofESinclude–

- KnowledgeBase
- InferenceEngine
- UserInterface

Letus seethemonebyonebriefly–



KnowledgeBase

It contains domain-specific and high-quality knowledge. Knowledge is required to exhibit intelligence.ThesuccessofanyESmajorlydependsuponthecollectionofhighlyaccurateand precise knowledge.

What is Knowledge?

The data is collection of facts. The information is organized as data and facts about the task domain.Data,information,andpastexperiencecombinedtogetheraretermedasknowledge.

Components of Knowledge Base

Theknowledgebaseof an ES is astoreof both,factual and heuristicknowledge.

- FactualKnowledge−ItistheinformationwidelyacceptedbytheKnowledgeEngineers and scholars in the task domain.
- HeuristicKnowledge−Itisaboutpractice,accuratejudgement,one'sabilityof evaluation, and guessing.

Knowledgerepresentation

Itisthemethodusedtoorganizeandformalizetheknowledgeintheknowledgebase. Itisinthe form of IF-THEN-ELSE rules.

KnowledgeAcquisition

Thesuccessofanyexpertsystemmajorlydependsonthequality,completeness,andaccuracyof the information stored in the knowledge base.

Theknowledgebaseisformedbyreadingsfromvariousexperts,scholars,andthe Knowledge Engineers. The knowledge engineer is a person with the qualities of empathy, quick learning, and case analyzing skills.

He acquires information from subject expert by recording, interviewing, and observing him at work,etc.Hethencategorizesandorganizestheinformationinameaningfulway,intheformof IF-THEN-ELSE rules, to be used by interference machine. The knowledge engineer also monitors the development of the ES.

InferenceEngine

Useofefficientproceduresandrulesbythe InferenceEngineisessentialindeductingacorrect, flawless solution.

Incaseofknowledge-basedES,theInferenceEngineacquiresandmanipulatestheknowledge from the knowledge base to arrive at a particular solution.

In caseofrulebasedES,it−

- Appliesrules repeatedlyto thefacts, whichareobtainedfrom earlierruleapplication.
- Addsnew knowledgeinto theknowledgebaseif required.
- Resolvesrulesconflictwhenmultiplerulesareapplicabletoaparticularcase. To

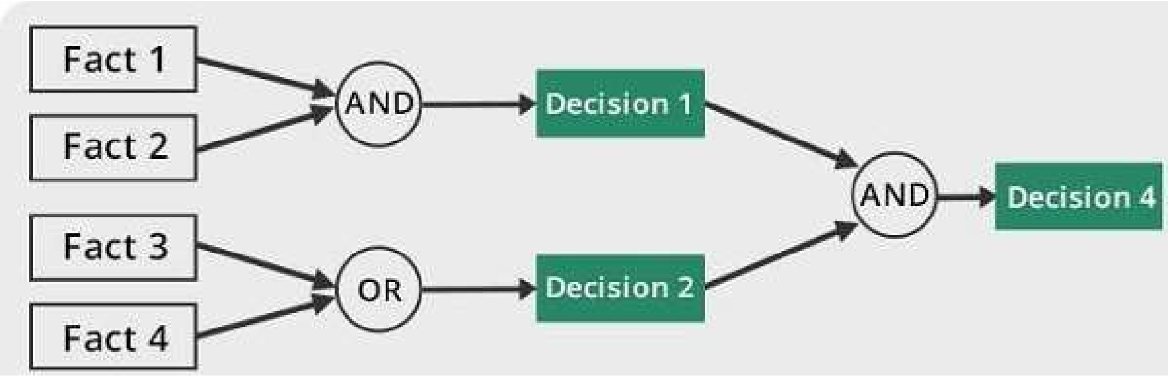
recommend a solution, the Inference Engine uses the following strategies −

- ForwardChaining
- BackwardChaining

Forward Chaining

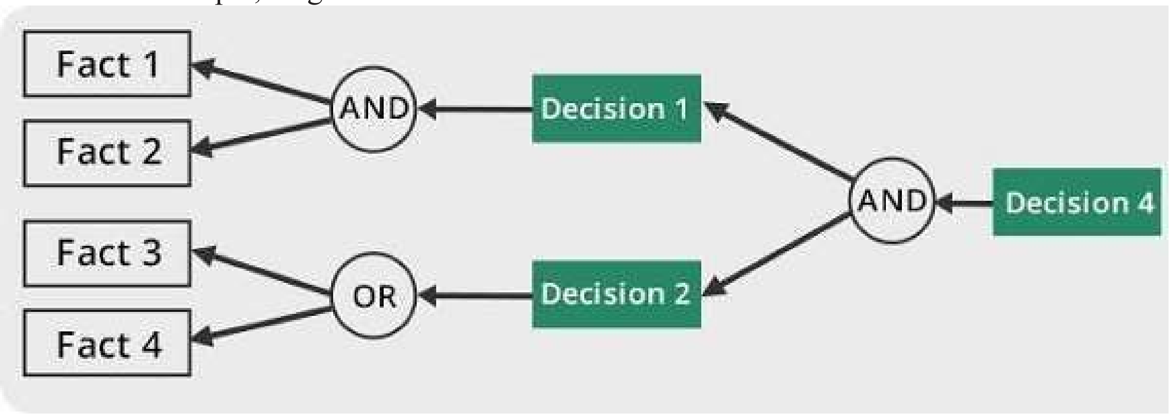
Itis astrategyof anexpert systemto answer thequestion,"What can happen next?"

Here, the Inference Engine follows the chain of conditions and derivations and finally deduces the outcome. It considers all the facts and rules, and sorts them before concluding to a solution. Thisstrategyisfollowedforworkingonconclusion,result,oreffect.Forexample,predictionof share market status as an effect of changes in interest rates.



BackwardChaining

Withthisstrategy,anexpertsystemfindsouttheanswertothequestion, "Whythishappened?" On the basis of what has already happened, the Inference Engine tries to find out which conditions could have happened in the past for this result. This strategy is followed for finding out cause or reason. For example, diagnosis of blood cancer in humans.



UserInterface

User interface provides interaction between user of the ES and the ES itself. It is generally Natural LanguageProcessingsoastobeusedbytheuserwhoiswell-versedinthetaskdomain. The user of the ES need not be necessarily an expert in Artificial Intelligence.

ItexplainshowtheEShasarrivedataparticularrecommendation. Theexplanationmayappear in the following forms –

- Naturallanguagedisplayedonscreen.
- Verbalnarrationsinnaturallanguage.
- Listingofrulenumbersdisplayedonthe screen.

Theuserinterfacemakesiteasytotracethecredibilityofthedeductions.

Requirements of Efficient ES User Interface

- Itshouldhelpuserstoaccomplish theirgoalsinshortestpossibleway.
- Itshouldbe designedto workforuser'sexistingor desiredwork practices.
- Itstechnologyshould be adaptableto user'srequirements;notthe otherwayround.
- Itshouldmakeefficientuseofuserinput.

Expert Systems Limitations

No technology can offer easy and complete solution. Large systems are costly, require significantdevelopmenttime,andcomputerresources.EShavetheirlimitationswhichinclude –

- Limitationsofthe technology

- Difficultknowledgeacquisition
- ESaredifficultto maintain
- Highdevelopmentcosts

Applications of Expert System

Thefollowingtable shows whereES can beapplied.

Application	Description
Design Domain	Cameralensdesign, automobile design.
MedicalDomain	DiagnosisSystemstodeducecauseofdiseasefromobserved data, conduction medical operations on humans.
MonitoringSystems	Comparingdatacontinuouslywithobservedsystemorwith prescribed behavior such as leakage monitoring in long petroleum pipeline.
ProcessControlSystems	Controllingaphysicalprocessbasedonmonitoring.
KnowledgeDomain	Findingoutfaultsinvehicles, computers.
Finance/Commerce	Detectionofpossiblefraud,suspicioustransactions,stock market trading, Airline scheduling, cargo scheduling.

ExpertSystemTechnology

ThereareseverallevelsofEStechnologiesavailable.Expertsystemstechnologiesinclude −

- ExpertSystemDevelopmentEnvironment−TheESdevelopmentenvironmentincludes hardware and tools. They are −
  - Workstations,minicomputers, mainframes.
  - HighlevelSymbolicProgrammingLanguagessuchas LIStProgramming(LISP) and PROgrammation en LOGique (PROLOG).
  - Largedatabases.
- Tools−Theyreducetheeffortandcostinvolvedindevelopinganexpertsystemtolarge extent.
  - Powerfuleditors and debuggingtools with multi-windows.
  - Theyproviderapidprototyping
  - HaveInbuiltdefinitionsofmodel,knowledgerepresentation,andinference design.
- Shells − A shell is nothing but an expert system without knowledge base. A shell providesthedeveloperswithknowledgeacquisition,inferenceengine,userinterface,and explanation facility. For example, few shells are given below −
  - JavaExpertSystemShell(JESS)thatprovidesfullydevelopedJavaAPIfor creating an expert system.
  - *Vidwan*, a shell developed at the National Centre for Software Technology, Mumbaiin1993.ItenablesknowledgeencodingintheformofIF-THENrules.

DevelopmentofExpertSystems:General Steps

TheprocessofESdevelopmentisiterative.StepsindevelopingtheESinclude− Identify Problem Domain

- Theproblemmustbe suitablefor anexpert systemtosolve it.
- Findtheexperts intask domainfortheES project.
- Establishcost-effectivenessofthesystem.

Design the System

- Identifythe ES Technology
- Knowandestablish thedegreeof integrationwiththe othersystemsand databases.
- Realizehowtheconceptscanrepresentthedomainknowledgebest.

Develop the Prototype

FromKnowledgeBase: Theknowledge engineerworksto−

- Acquiredomainknowledge fromthe expert.
- RepresentitintheformofIf-THEN-ELSErules.

Test and Refine the Prototype

- Theknowledgeengineerusessamplecasestotesttheprototypeforanydeficienciesin performance.
- EnduserstesttheprototypesoftheES.

Develop and Complete the ES

- TestandensuretheinteractionoftheESwithallelementsofitsenvironment,including end users, databases, and other information systems.
- DocumenttheESprojectwell.
- TraintheusertouseES.

Maintain the ES

- Keeptheknowledgebaseup-to-date byregular reviewand update.
- Caterfornewinterfaceswithotherinformationsystems,asthosesystemsevolve.

Benefits of Expert Systems

- Availability−Theyareeasilyavailabledueto mass production of software.
- LessProductionCost −Productioncostisreasonable.Thismakesthemaffordable.
- Speed−Theyoffergreatspeed. Theyreducetheamount ofwork anindividual puts in.
- LessErrorRate−Errorrate islowascomparedtohuman errors.
- ReducingRisk−Theycan work in theenvironment dangerous to humans.
- Steadyresponse−Theywork steadilywithout gettingmotional, tensed orfatigued.

Expert System.

DEFINITION - An expert system is a computer program that simulates the judgement and behavior of a human or an organization that has expert knowledge and experience in a particular field. Typically, such a system contains a knowledge base containing accumulated experience andasetofrulesforapplyingtheknowledgebase toeachparticularsituationthatisdescribedto theprogram.Sophisticatedexpertsystemscanbeenhancedwithadditionstotheknowledgebase or to the set of rules.

Amongthebest-knownexpertsystemshavebeenthosethatplaychessandthatassistinmedical diagnosis.

An **expert system** is softwarethat attempts to provide an answer to a problem, or clarify uncertainties where normally one or more human expertswould need to be consulted. Expert systemsaremostcommoninaspecificproblemdomain,andisatraditionalapplicationand/or

subfield of artificial intelligence (AI). A wide variety of methods can be used to simulate the performanceoftheexpert;however,commontomostorallare:1)thecreationofa knowledge base which uses some knowledge representation structure to capture the knowledge of the Subject Matter Expert (SME); 2) a process of gatheringthat knowledge from the SME and codifyingit accordingto the structure, which is called knowledgeengineering; and 3) oncethe systemisdeveloped,itisplacedinthesamerealworld problemsolvingsituationasthehuman SME, typically as an aid to human workers or as a supplement to some information system.
Expert systems mayormaynot havelearning components.

factors

TheMYCINrule-basedexpertsystemintroducedaquasi-probabilisticapproachcalledcertainty factors, whose rationale is explained below.

A human, when reasoning, does not always make statements with 100% confidence: he might venture, "If Fritz is green, then he is probably a frog" (after all, he might be a chameleon). This type of reasoning can be imitated using numeric values called confidences. For example, if it is known that Fritz is green, it might be concluded with 0.85 confidence that he is a frog; or, if it is known that he is afrog, it might be concluded with 0.95 confidencethat he hops. Thesecertainty factor (CF) numbers quantify uncertainty in the degree to which the available evidence supportsa hypothesis. They represent a degree of confirmation, and are not probabilities in a Bayesian sense. The CF calculus, developed by Shortliffe & Buchanan, increases or decreases the CF associated with a hypothesis as each new piece of evidence becomes available. It can be mapped to a probability update, although degrees of confirmation are not expected to obey the laws of probability.Itisimportanttonote,forexample,thatevidenceforhypothesisHmayhavenothing to contribute to the degree to which Not_h is confirmed or disconfirmed (e.g., although a fever lends some support to a diagnosis of infection, fever does not disconfirm alternative hypotheses) and that the sum of CFs of many competing hypotheses may be greater than one (i.e., many hypotheses may be well confirmed based on available evidence).

TheCFapproachtoarule-basedexpertsystemdesigndoesnothaveawidespreadfollowing,in part because of the difficulty of meaningfully assigning CFs a priori. (The above example of green creatures being likely to be frogs is excessively naive.) Alternative approaches to quasi-probabilistic reasoning in expert systems involve fuzzylogic, which has a firmer mathematical foundation. Also, rule-engine shells such as Drools and Jess do not support probability manipulation: theyuse an alternative mechanism called salience, which is used to prioritize the order of evaluation of activated rules.

In certainareas,asinthe tax-advicescenariosdiscussedbelow,probabilisticapproaches arenot acceptable. For instance, a 95% probability of being correct means a 5% probability of being wrong. The rules that are defined in such systems have no exceptions: they are onlya means of achievingsoftwareflexibilitywhenexternalcircumstanceschangefrequently.Becauserulesare stored as data, the core software does not need to be rebuilt each time changes to federal and state tax codes are announced.

Chaining

Twomethodsofreasoningwhenusinginferencerulesareforwardchainingandbackward chaining.

Forwardchainingstartswiththedataavailableandusesthe inferencerulestoextractmoredata until a desired goal is reached. An inference engine using forward chaining searches the inferencerulesuntilitfindsoneinwhichtheifclauseisknowntobetrue.Itthenconcludesthe then clause and adds this information to its data. It continues to do this until a goal is reached. Because the data available determines which inference rules are used, this method is also classified as data driven.

Backward chaining starts with a list of goals and works backwards to see if there is data which will allow it to conclude anyof these goals. An inference engine usingbackward chainingwould searchtheinferencerulesuntilitfindsonewhichhasathenclausethatmatchesadesiredgoal.If the if clause of that inference rule is not known to be true, then it is added to the list of goals.

SW Architecture.

Thefollowinggeneralpointsaboutexpertsystemsandtheirarchitecturehavebeen outlined:

1. Thesequenceofstepstakentoreachaconclusionisdynamicallysynthesizedwitheachnew case. The sequence is not explicitly programmed at the time that the system is built.

2. Expertsystemscanprocessmultiplevaluesforanyproblemparameter.Thispermitsmore than oneline of reasoningto be pursued and the results of incomplete (not fullydetermined) reasoning to be presented.

3. Problem solving is accomplished by applying specific knowledge rather than specific technique. This is a key idea in expert systems technology. It reflects the belief that human experts do not process their knowledge differently from others, but they do possess different knowledge.Withthisphilosophy,whenonefindsthattheirexpertsystemdoesnotproducethe desired results, work begins to expand the knowledge base, not to re-program the procedures.

Enduser

Therearetwostylesofuser-interfacedesignfollowedbyexpertsystems.Intheoriginalstyleof user interaction, the software takes the end-user through an interactive dialog. In the following example, a backward-chaining system seeks to determine a set of restaurants to recommend:

Q.Do youknowwhichrestaurantyouwanttogoto?

A. No

Q.Isthere anykind offoodyou wouldparticularlylike?

A. No

Q.Doyou likespicyfood?

A. No

Q.Doyou usuallydrink wine with meals?

A. Yes

Q.Whenyoudrinkwine, isitFrenchwine?

A. Yes

Participants

There are generally three individuals having an interaction in an expert system. Primary among these is the end-user, the individual who uses the system for its problem solving assistance. Inthe construction and maintenance of the system there are two other roles: the problem domain expert who builds the system and supplies the knowledge base, and a knowledge engineer who assists the experts in determining the representation of their knowledge, enters this knowledge into an explanation module and who defines the inference technique required to solve the problem. Usuallythe knowledge engineer will represent the problem solving activityin the form ofrules.Whentheserulesarecreatedfromdomainexpertise,theknowledgebasestorestherules of the expert system.

Inference rule

An understanding of the "inference rule" concept is important to understand expert systems. An inference rule is a conditional statement with two parts: an if clause and a then clause. This rule iswhatgivesexpertsystemstheabilitytofindsolutionstodiagnosticandprescriptiveproblems. An example of an inference rule is:

IftherestaurantchoiceincludesFrenchandtheoccasionisromantic, Then the
restaurant choice is definitely Paul Bocuse.

Procedurenodeinterface

The function of the procedure node interface is to receive information from the procedures coordinatorandcreatetheappropriateprocedurecall.Theabilitytocallaprocedureandreceive information from that procedure can be viewed as simply a generalization of input from the external world. In some earlier expert systems external information could only be obtained in a predetermined manner, which only allowed certain information to be acquired. Through the knowledge base, this expert system disclosed in the cross-referenced application can invoke any procedureallowedonitshostsystem.Thismakestheexpertsystemusefulinamuchwiderclass of knowledge domains than if it had no external access or only limited external access.

Inthe areaofmachinediagnosticsusingexpertsystems,particularlyself-diagnostic applications, itisnotpossibletoconcludethecurrentstateof"health"ofamachinewithoutsomeinformation. The best source of information is the machine itself, for it contains much detailed information that could not reasonably be provided by the operator.

The knowledge that is represented in the system appears in the rulebase. In the rulebase describedinthecross-referencedapplications,therearebasicallyfourdifferenttypesofobjects, with the associated information:

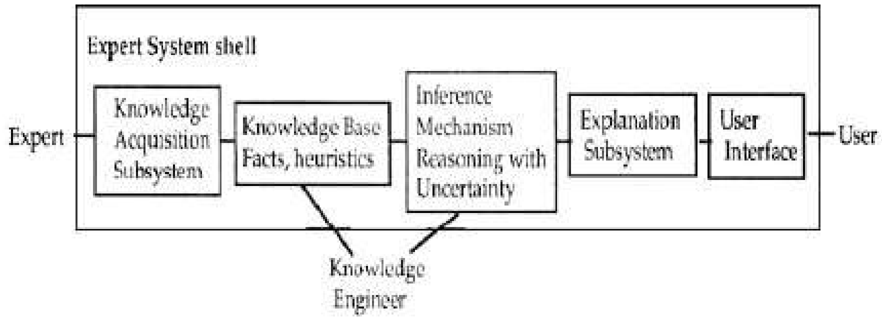
1. Classes:Questionsaskedtotheuser.

2. Parameters:Placeholdersforcharacterstringswhichmaybevariablesthatcanbe inserted into a class question at the point in the question where the parameter is positioned.

3. Procedures:Definitionsofcallstoexternalprocedures.

3. Rule nodes: Inferences in the system are made by a tree structure which indicates the rulesorlogicmimickinghumanreasoning.Thenodesofthesetreesarecalledrulenodes. There are several different types of rule nodes.

Expert Systems/Shells. The E.S **shell** simplifies the process of creating a knowledge base. It is the **shell** that actually processes the information entered by a user relates it to the concepts containedintheknowledgebaseandprovidesanassessmentorsolutionforaparticularproblem.


Expert System shell
Expert
Knowledge Acquisition Subsystem
Knowledge Base Facts, heuristics
Inference Mechanism Reasoning with Uncertainty
Explanation Subsystem
User Interface
User
Knowledge Engineer

KnowledgeAcquisition

Knowledge acquisition is the process used to define the rules and ontologies required foraknowledge-basedsystem.Thephrasewasfirstusedinconjunctionwithexpertsystemsto describe the initial tasks associated with developing an expert system, namely finding and interviewing domain experts and capturing their knowledge via rules, objects, and frame-based ontologies.

Expertsystemswereoneofthefirstsuccessfulapplicationsof artificialintelligencetechnology torealworldbusinessproblems.ResearchersatStanfordandotherAIlaboratoriesworkedwith doctors and other highly skilled experts to develop systems that could automate complex tasks such as medical diagnosis. Until this point computers had mostlybeen used to automate highly data intensive tasks but not for complex reasoning. Technologies such as inference enginesalloweddevelopersforthefirsttimetotacklemorecomplexproblems.
Asexpertsystemsscaledupfromdemonstrationprototypestoindustrialstrengthapplicationsit was soon realized that the acquisition of domain expert knowledge was one of if not the most critical task in the knowledge engineering process. This knowledgeacquisition process became an intense area of research on its own. One of the earlier works on the topic used Batesonian theories of learning to guide the process.
One approach to knowledge acquisition investigated was to use natural language parsing and generation to facilitate knowledge acquisition. Natural language parsing could be performed on manuals and other expert documents and an initial first pass at the rules and objects could be developed automatically. Text generation was also extremely useful in generating explanations forsystembehavior.Thisgreatlyfacilitatedthedevelopmentandmaintenanceofexpertsystems. A more recent approach to knowledge acquisition is a re-use based approach. Knowledge can be developed in ontologies that conform to standards such as the Web Ontology Language (OWL). Inthiswayknowledgecanbestandardizedandsharedacrossabroadcommunityof knowledge workers. One example domain where this approach has been successful
is bioinformatics.

Refferences

1. ElaineRich,KevinKnight,&ShivashankarBNair,ArtificialIntelligence, McGrawHill,3rded.,2009 References:
1) IntroductiontoArtificialIntelligence&ExpertSystems,DanWPatterson, PHI.,2010

2) SKaushik,ArtificialIntelligence,CengageLearning,1sted.2011