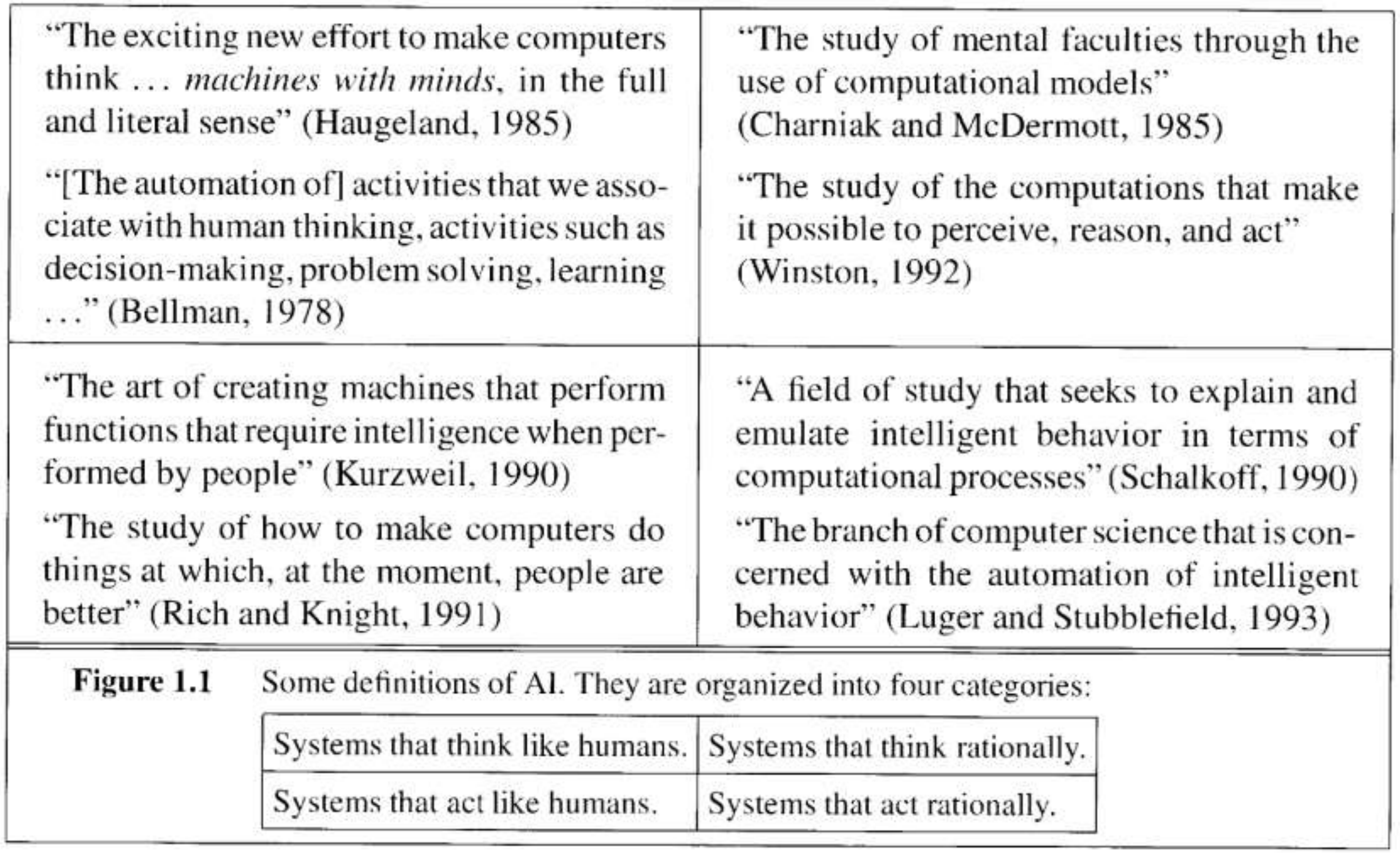
18CSC305J ARTIFICIAL INTELLIGENCE

Introduction

What is AI?



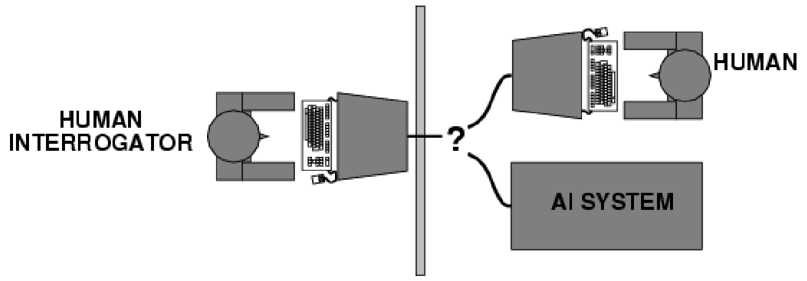
"The exciting new effort to make computers think … *machines with minds*, in the full and literal sense" (Haugeland, 1985) "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning …" (Bellman, 1978)	"The study of mental faculties through the use of computational models" (Charniak and McDermott, 1985) "The study of the computations that make it possible to perceive, reason, and act" (Winston, 1992)
"The art of creating machines that perform functions that require intelligence when performed by people" (Kurzweil, 1990) "The study of how to make computers do things at which, at the moment, people are better" (Rich and Knight, 1991)	"A field of study that seeks to explain and emulate intelligent behavior in terms of computational processes" (Schalkoff, 1990) "The branch of computer science that is concerned with the automation of intelligent behavior" (Luger and Stubblefield, 1993)

Figure 1.1 Some definitions of AI. They are organized into four categories:

Systems that think like humans.	Systems that think rationally.
Systems that act like humans.	Systems that act rationally.

Acting humanly: Turing Test

- Turing (1950) "Computing machinery and intelligence":
- "Can machines think?" → "Can machines behave intelligently?"
- Operational test for intelligent behavior: the Imitation Game



The computer would need to possess the following capabilities:

- **natural language processing** to enable it to communicate successfully in English (or some other human language);
- **knowledge representation** to store information provided before or during the interrogation;
- **automated reasoning** to use the stored information to answer questions and to draw new conclusions;
- **machine learning** to adapt to new circumstances and to detect and extrapolate patterns.

To pass the total Turing Test, the computer will need

- **computer vision** to perceive objects
- **robotics** to move them about.

Thinking humanly: cognitive modeling

Determining how humans think

- through introspection—trying to catch our own thoughts as they go by
- through psychological experiments

Express the theory as a computer program

- program's input/output and timing behavior matches human behavior

Thinking rationally: "laws of thought"

- Aristotle: what are correct arguments/thought processes?
- Several Greek schools developed various forms of *logic*: *notation* and *rules of derivation* for thoughts; may or may not have proceeded to the idea of mechanization
- Direct line through mathematics and philosophy to modern AI
- Problems:
  1. Not all intelligent behavior is mediated by logical deliberation
  2. What is the purpose of thinking? What thoughts should I have?

Acting rationally: rational agent

- Rational behavior: doing the right thing
- The right thing: that which is expected to maximize goal achievement, given the available information
- An **agent** is just something that perceives and acts
- Doesn't necessarily involve thinking – but thinking should be in the service of rational action

Rational agents

- An agent is an entity that perceives and acts
- Abstractly, an agent is a function from percept histories to actions:

$$[f: \mathcal{P}^\star \rightarrow \mathcal{A}]$$

- For any given class of environments and tasks, we seek the agent (or class of agents) with the best performance
- computational limitations make perfect rationality unachievable

  → design best program for given machine resources

History of AI

- AI has roots in a number of scientific disciplines
  - computer science and engineering (hardware and software)
  - philosophy (rules of reasoning)
  - mathematics (logic, algorithms, optimization)
  - cognitive science and psychology (modeling high level human/animal thinking)
  - neural science (model low level human/animal brain activity)
  - linguistics
- The birth of AI (1943 – 1956)
  - McCulloch and Pitts (1943): simplified mathematical model of neurons (resting/firing states) can realize all propositional logic primitives (can compute all Turing computable functions)
  - Alan Turing: Turing machine and Turing test (1950)
  - Claude Shannon: information theory; possibility of chess playing computers
  - Boole, Aristotle, Euclid (logics, syllogisms)

History of AI

- Early enthusiasm (1952 – 1969)
  - 1956 Dartmouth conference
    John McCarthy (Lisp);
    Marvin Minsky (first neural network machine);
    Alan Newell and Herbert Simon (GPS);
  - Emphasis on intelligent general problem solving
    GSP (means-ends analysis);
    Lisp (AI programming language);
    Resolution by John Robinson (basis for automatic theorem proving);
    heuristic search (A*, AO*, game tree search)
- Emphasis on knowledge (1966 – 1974)
  - domain specific knowledge is the key to overcome existing difficulties
  - knowledge representation (KR) paradigms
  - declarative vs. procedural representation

History of AI

- Knowledge-based systems (1969 – 1999)
  - DENDRAL: the first knowledge intensive system (determining 3D structures of complex chemical compounds)
  - MYCIN: first rule-based expert system (containing 450 rules for diagnosing blood infectious diseases)
    EMYCIN: an ES shell
  - PROSPECTOR: first knowledge-based system that made significant profit (geological ES for mineral deposits)
- AI became an industry (1980 – 1989)
  - wide applications in various domains
  - commercially available tools
  - AI winter
- Current trends (1990 – present)
  - more realistic goals
  - more practical (application oriented)
  - distributed AI and intelligent software agents
  - resurgence of natural computation - neural networks and emergence of genetic algorithms – many applications
  - dominance of machine learning (big apps)

State of the art

- HITECH, becomes the first computer program to defeat a grandmaster in a game of chess (Arnold Denker)
- A speech understanding program named PEGASUS results in a confirmed reservation that saves the traveller $894 over the regular coach fare.
- MARVEL, a real-time expert system that monitors the massive stream of data transmitted by the spacecraft, handling routine tasks and alerting the analysts to more serious problems.

Advantages of Artificial Intelligence

- more powerful and more useful computers
- new and improved interfaces
- solving new problems
- better handling of information
- relieves information overload
- conversion of information into knowledge

The Disadvantages

- increased costs
- difficulty with software development - slow and expensive
- few experienced programmers
- few practical products have reached the market as yet.

AI Technique

- AI deals with a large spectrum of Problems
- Applications spread across the domains, from medical to manufacturing with their own complexities
- AI Deals with
  - Various Day-to-day Problem
  - Different identification and authentication problems (in security)
  - Classification problems in Decision-making systems
  - Interdependent and cross-domain problems (Such as Cyber-Physical
  - Systems)
- The problems faced by AI is hard to resolve and also computationally

AI Technique

- Intelligence requires knowledge

  (less desirable properties)

  – voluminous
  – hard to characterize accurately
  – constantly changing
  – differ from data by being organized in a way that corresponds to the ways it will be used

Knowledge Representation

- Generalizations-defines property
- Understood by people -eg taking readings
- Easily modified – correct errors and reflect changes
- Used in a great many situations(even not accurate or complete)
- Can be used to reduce the possibilities that must be considered(bulk to narrow)

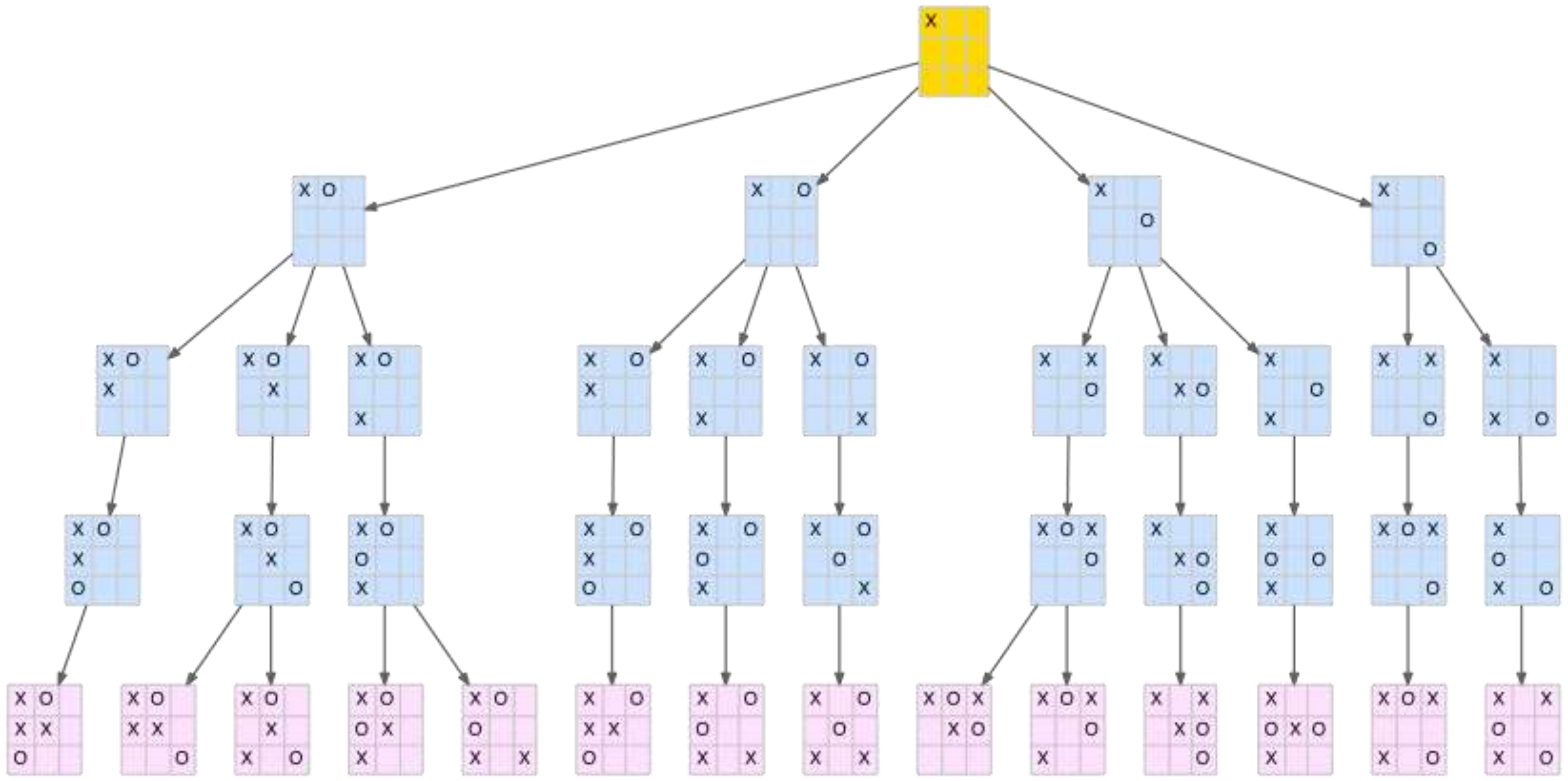
Categories of problems

- Structured problems –goal state defined
- Unstructured problems- goal state not known
- Linear problems- based on dependent variable
- Non linear problems- no dependency between variables

Problem

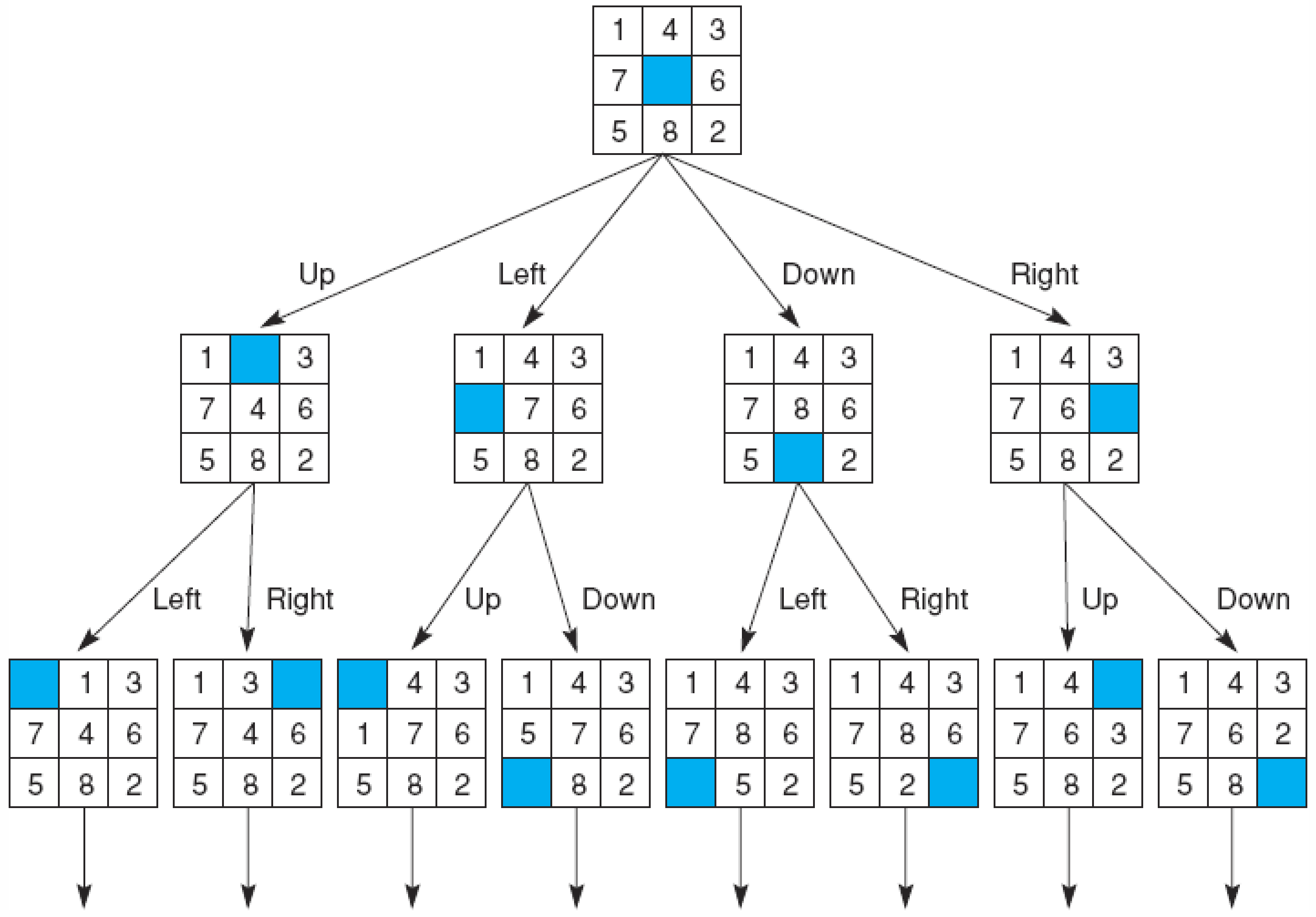
- In AI, formally define a problem as
  - a space of all possible configurations where each configuration is called a state
  - The ***state-space*** is the configuration of the possible states and how they connect to each other e.g. the legal moves between states.
  - an initial state
  - one or more goal states
  - a set of rules/operators which move the problem from one state to the next
- In some cases, we may enumerate all possible states
  - but usually, such an enumeration will be overwhelmingly large so we only generate a portion of the state space, the portion we are currently examining
  - we need to search the state-space to find an optimal path from a start state to a goal state.

State space: Tic-Tac-Toe



Goal: Arrange in horizontal or vertical or diagonal to win

State space: 8 Puzzle



The 8 puzzle search space consists of 8! states (40320)

Search

- Search is a general algorithm that helps in finding the path in state space
- The path may lead to the solution or dead end.
- Control strategies- overall rules and approach towards searching

  i) forward search(data directed)

    Starts search from initial state towards goal state.

    Ex: locating a city from current location

  ii) backward search(goal directed)

    Search stars from goal state towards a solvable initial state.

    Ex: start from target city

Search

- Strategies to explore the states
  - Informed search – No guarantee for solution but high probability of getting solution
    -heuristic approach is used to control the flow of solution path
    -heuristic approach is a technique based on common sense, rule of thumb, educated guesses or intuitive judgment
  - Uninformed search – generates all possible states in the state space and checks for the goal state.
    - time consuming due to large state space
    - used where error in the algorithm has severe consequences

- Parameters for search evaluation
  i) completeness: Guaranteed to find a solution within finite time
  ii) space and time complexity: memory required and time factor needed
  iii) optimality and admissibility: correctness of the solution

Problem solving with AI

Well structured problems- yield right answer or inference for an algorithm

Ex:

- Quadratic equation-find value of x
- Speed of ball when reaches to batsman
- Network flow analysis

Ill structured problems-do not yield a particular answer

Ex:

- How to dispose wet waste safely
- Security threats in big social gathering

Unstructured problems- exact goal state not known(many goal states)

Ex: improve life expectancy of human being

Linear problems-have a solution or will not have

Non Linear problems-relationship between input and output are not linear

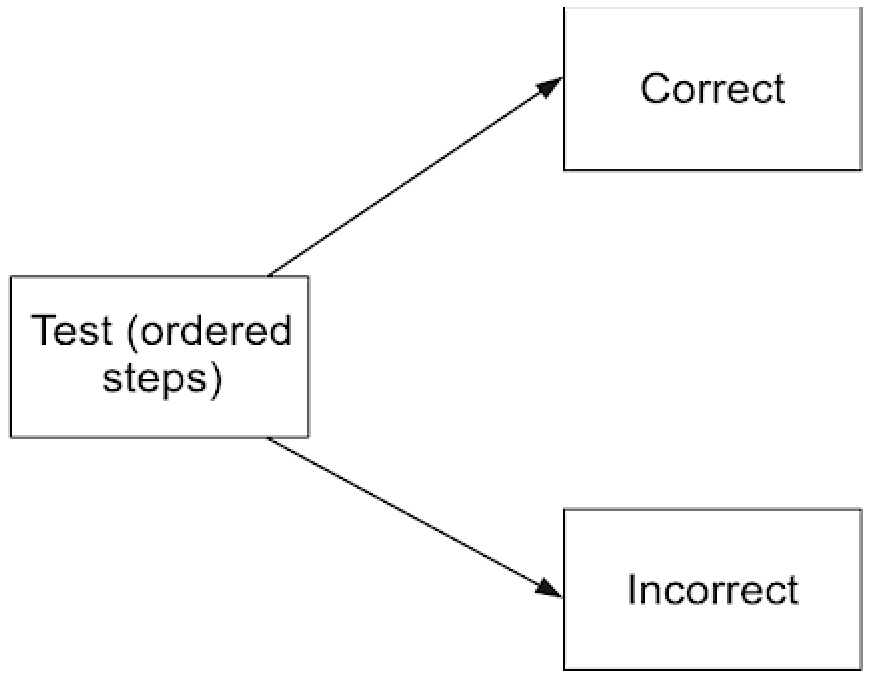


Figure 1.2 A typical well-structured problem analysis.

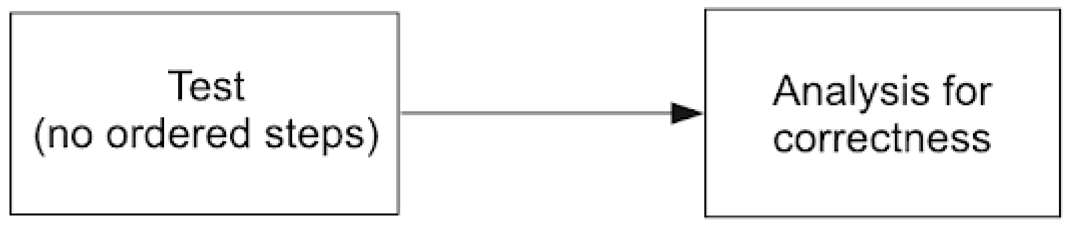


Figure 1.3 A typical ill-structured problem analysis.

AI Applications

- Credit granting
- Information management and retrieval
- AI and expert systems embedded in products
- Plant layout
- Help desk and assistance
- Employee performance evaluation
- Shipping
- Marketing
- Warehouse optimization
- In space workstation maintainance
- Satellite controls
- Network developments
- Nuclear management

AI Models

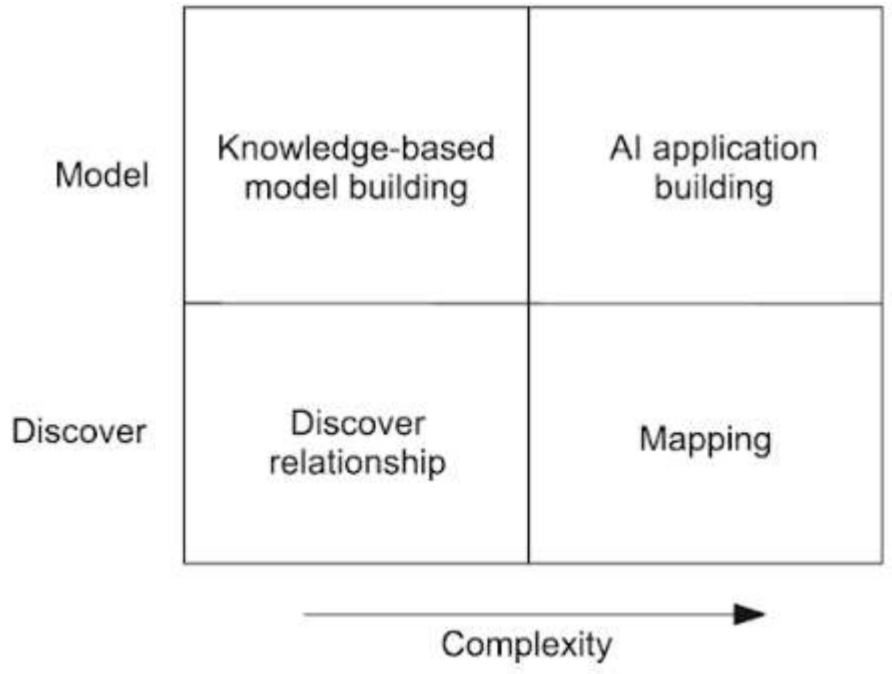


Figure 1.5 Model building and complexity.

Semiotic models- Based on sign process, signification or communication
Statistical models- representation and formalization of relationships through statistical techniques.
- History of data for decision making
- uses probabilistic approaches

Data Acquisition and Learning Aspects in AI

- Knowledge Discovery- data mining and machine learning:

  data-recorded facts

  information-pattern underlying the data

  data mining or knowledge discovery-extraction of meaning information.

  machine learning-algorithms that improve performance with experience

- Computational Learning Theory(COLT)- formal mathematical models defined

  complexity-computation, prediction and feasibility

  analyze patterns-Probably Approximately Correct(PAC)-hypothesis

  mistake bound-target function

- Neural and evolutionary computation- speed up mining of data

  evolutionary computing- biological properties

  decision making and optimization

  Neural computing-neural behavior of human being

  pattern recognition and classification

Data Acquisition and Learning Aspects in AI

- Intelligent agents and multi agent systems- decision making in complex scenarios

Intelligent agents –based on knowledge, available resources and perspectives

multi agent systems- combination of more than one percept of intelligent agents

- Multi-perspective integrated intelligence-utilizing and exploiting knowledge from different perspective

Problem Solving: Definition

A problem exists when you want to get from “here” (a knowledge state) to “there” (another knowledge state) and the path is not immediately obvious.

Makes optimal use of knowledge and information to select set of actions for reaching the goal.

What are problems?

- ■Everyday experiences
  - ■How to get to the airport?
  - ■How to study for a quiz, complete a paper, and finish a lab before recitation?
- ■Domain specific problems
  - ■Physics or math problems
- ■Puzzles/games
  - ■Crossword, anagrams, chess

Categories of problem solving

- General purpose: means-end analysis

  present situation is compared with the goal to detect the difference

  select action that reduces the difference

  Ex:select the mode of transport
- Special purpose-modelled for the specific problem, which have specific features

  Ex: classify legal document reference to particular case

Problem solving process

- Problem solving-process of generating solutions for the given situation
- Problem is defined,
    1. in a context
    2. has well defined objective
    3. solution has set of activities
    4. uses previous knowledge and domain knowledge

Primary objective-problem identification

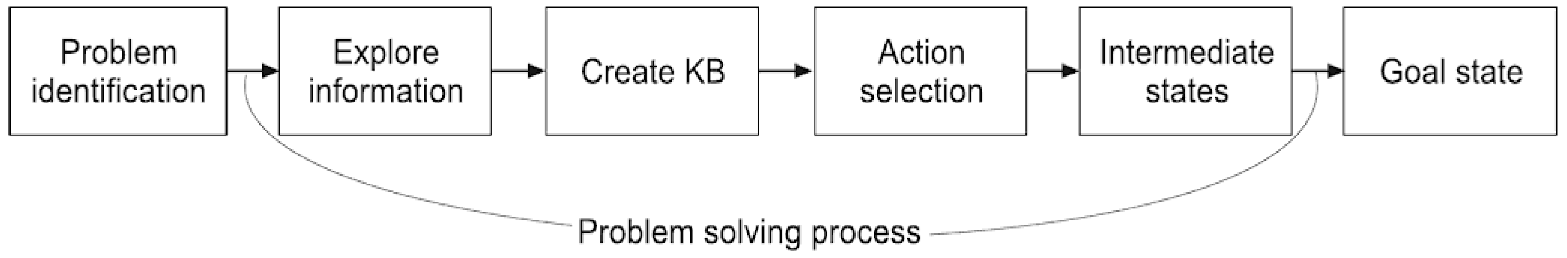


Figure 2.1 Problem-solving process.

Problem solving process

- Problem solving technique involves
  1. problem definition
  2. problem analysis and representation
  3. planning
  4. execution
  5. evaluating solution
  6. consolidating gains
- A search algorithm takes a problem as input and returns a **solution** in the form of an action sequence.
- **execution** phase-Once a solution is found, the actions it recommends can be carried out.

Formulating problems

- **Problem formulation** is the process of deciding what actions and states to consider, and follows goal formulation.
- **Goal formulation-**the agent may wish to decide on some other factors that affect the desirability of different ways of achieving the goal.
- **Knowledge and problem types**

single-state problem-Agent knows exactly what each of its actions does and it can calculate exactly which state it will be in after any sequence of actions.

multiple-state problem-when the world is not fully accessible, the agent must reason about sets of states that it might get to, rather than single states.

contingency problem-the agent may be in need to now calculate a whole tree of actions, rather than a single action sequence in which each branch of the tree deals with a possible contingency that might arise.

exploration problem-the agent learns a "map" of the environment, which it can then use to solve subsequent problems.

Formulating problems

- **Well-defined problems and solutions**

  A problem is really a collection of information that the agent will use to decide what to do.

  Elements of a problem:

  1. **The initial state** that the agent knows itself to be in.

  2. The set of possible actions available to the agent.

     operator is used to denote the description of an action to reach a state.

     state space-the set of all states reachable from the initial state by any sequence of actions.

     A **path** in the state space is simply any sequence of actions leading from one state to another.

  3. **The goal test,** which the agent can apply to a single state description to determine if it is a goal state.

  4. **A path cost** function is a function that assigns a cost to a path.

The output of a search algorithm is a **solution,** that is, a path from the initial state to a state that satisfies the goal test.

Formulating problems

- **Measuring problem-solving performance**

  Solution is obtained or not

  Obtained solution is good solution or not(with a low path cost)

  Search cost-associated with the time and memory required to find a solution.

  total cost of the search is the sum of the path cost and the search cost

- **Choosing states and actions**

  To decide a better solution, determine the measurement of path cost function

  The process of removing detail from a representation is called **abstraction**

Problem Formulation and Representation

- Example: Water jug problem-find out a way to empty 2 galloon jug and fill 5 galloon jug with 1 galloon of water

  states: Amount of water in the jugs

  actions: 1. empty the big jug

  2. empty the small jug

  3. pour water from small jug to big jug

  4. . pour water from big jug to small jug

  Goal: 1 galloon of water in big jug and empty the small jug

  path cost: number of actions(minimum number of actions->better solution)

  Representation: jugs(b,s), where b-amount of water in bigger jug, s- b-amount of water in smaller jug

  initial state: (5,2)

  goal state: (1,0)

  operators: i) empty the jug

  ii) fill the jug

Problem Formulation and Representation

- Solution:

  initial state: (5,2)

  goal state: (1,0)

  operators:

  i) empty big(remove water from big jug)

  ii) empty small(remove water from small jug)

  iii) big is empty(pour water from small jug to big jug)

  iv) small is empty(pour water from big jug to small jug)

  actions of sequence: 2,4,2,4,2

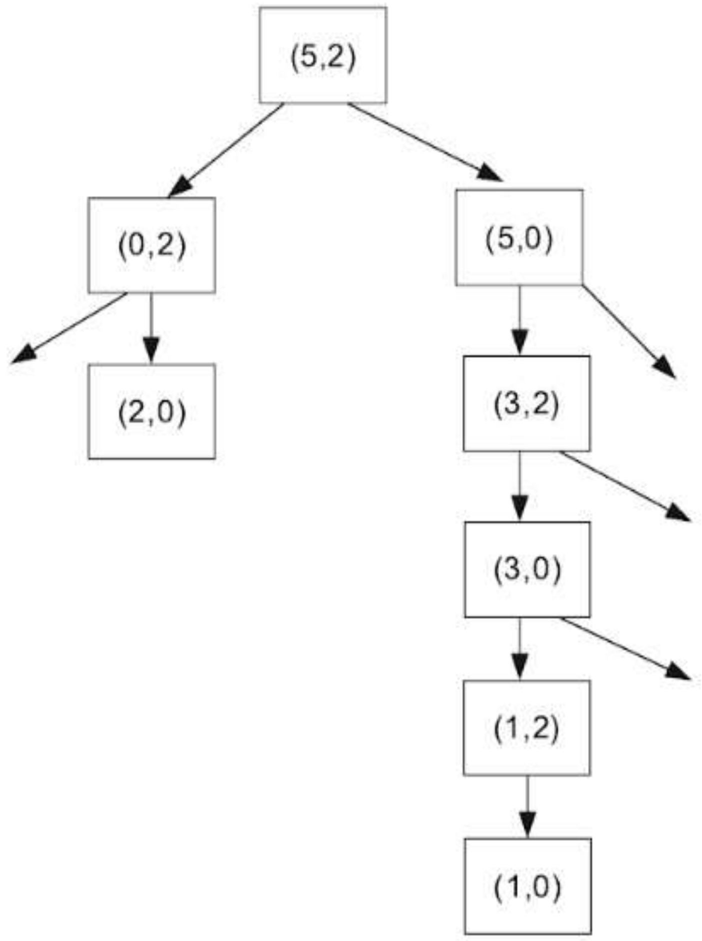
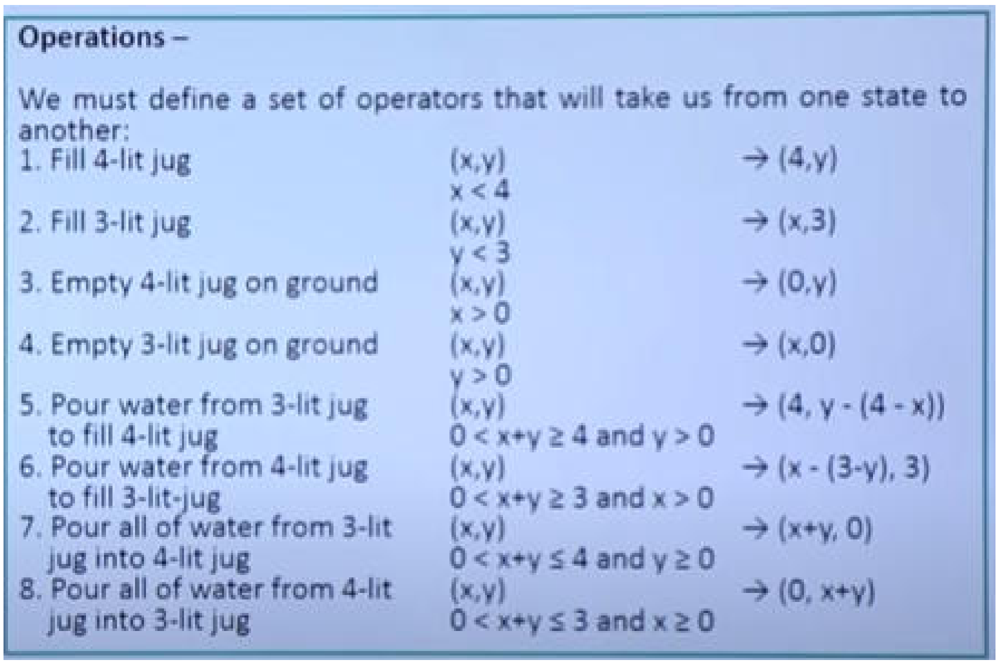


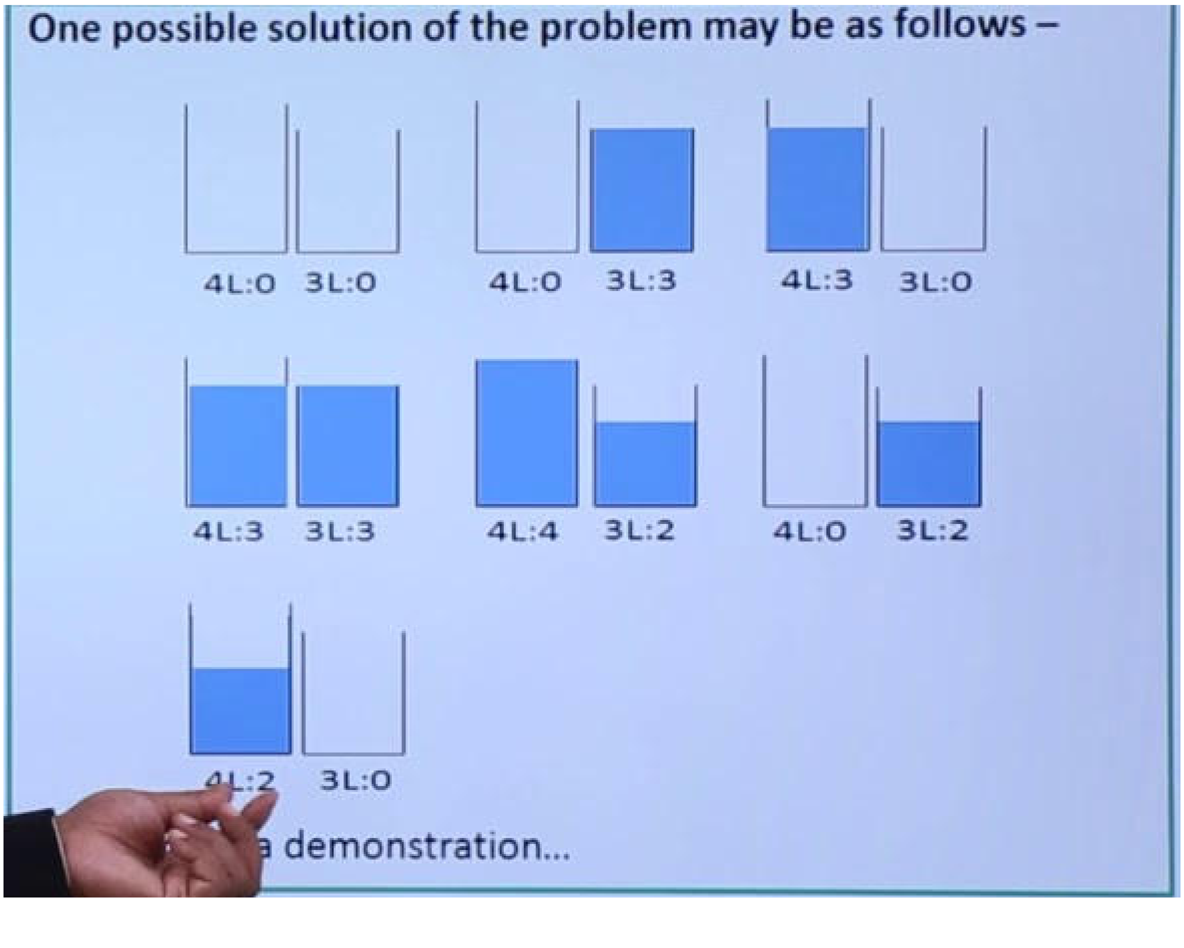
Figure 2.7 Water jug problem.



Operations –

We must define a set of operators that will take us from one state to another:

1. Fill 4-lit jug	(x,y) $x < 4$	$\rightarrow (4,y)$
2. Fill 3-lit jug	(x,y) $y < 3$	$\rightarrow (x,3)$
3. Empty 4-lit jug on ground	(x,y) $x > 0$	$\rightarrow (0,y)$
4. Empty 3-lit jug on ground	(x,y) $y > 0$	$\rightarrow (x,0)$
5. Pour water from 3-lit jug to fill 4-lit jug	(x,y) $0 < x+y \geq 4$ and $y > 0$	$\rightarrow (4, y - (4 - x))$
6. Pour water from 4-lit jug to fill 3-lit-jug	(x,y) $0 < x+y \geq 3$ and $x > 0$	$\rightarrow (x - (3-y), 3)$
7. Pour all of water from 3-lit jug into 4-lit jug	(x,y) $0 < x+y \leq 4$ and $y \geq 0$	$\rightarrow (x+y, 0)$
8. Pour all of water from 4-lit jug into 3-lit jug	(x,y) $0 < x+y \leq 3$ and $x \geq 0$	$\rightarrow (0, x+y)$



One possible solution of the problem may be as follows –

a demonstration...

Toy problems

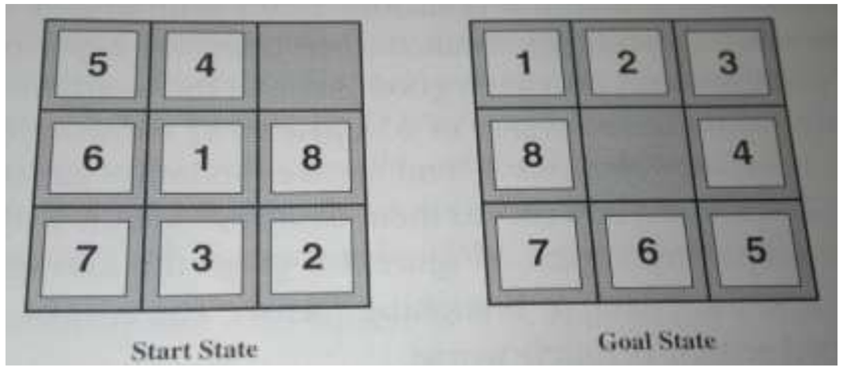
Intended to illustrate or exercise various problem-solving methods

The 8-puzzle

- 3x3 board with eight numbered tiles and a blank space.
- A tile adjacent to the blank space can slide into the space.
- objective-to reach the configuration shown on the right of the figure.

Problem formulation:

- States: a state description specifies the location of each of the eight tiles in one of the nine squares. For efficiency, it is useful to include the location of the blank.
- **Operators:** blank moves left, right, up, or down.
- **Goal test:** state matches the goal configuration shown in Figure.
- **Path cost:** each step costs 1, so the path cost is just the length of the path.



Toy problems

The 8-queens problem

- Place eight queens on a chessboard such that no queen attacks any other.
- There are two main kinds of formulation

  The *incremental* formulation involves placing queens one by one

  the *complete-state* formulation starts with all 8 queens on the board and moves them around.

- **Goal** test: 8 queens on board, none attacked.
- **Path** cost: zero.

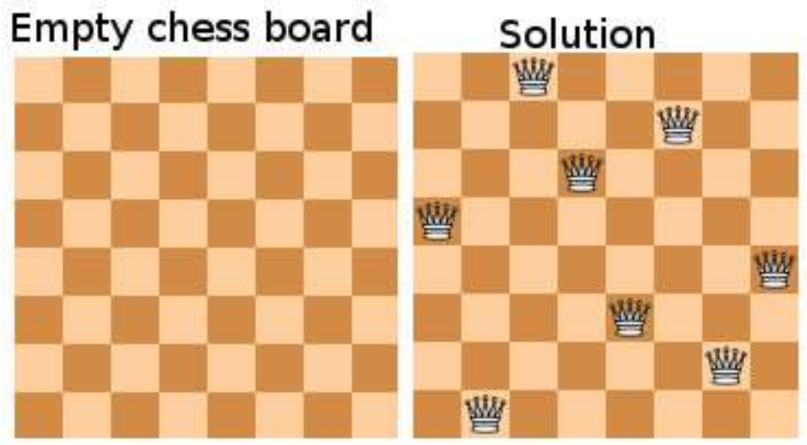
There are also different possible states and operators.

Consider the following for incremental formulation:

- **States:** any arrangement of 0 to 8 queens on board.
- **Operators:** add a queen to any square.

Consider the following for complete state formulation:

- **States:** arrangements of 8 queens, one in each column.
- **Operators:** move any attacked queen to another square in the same column.



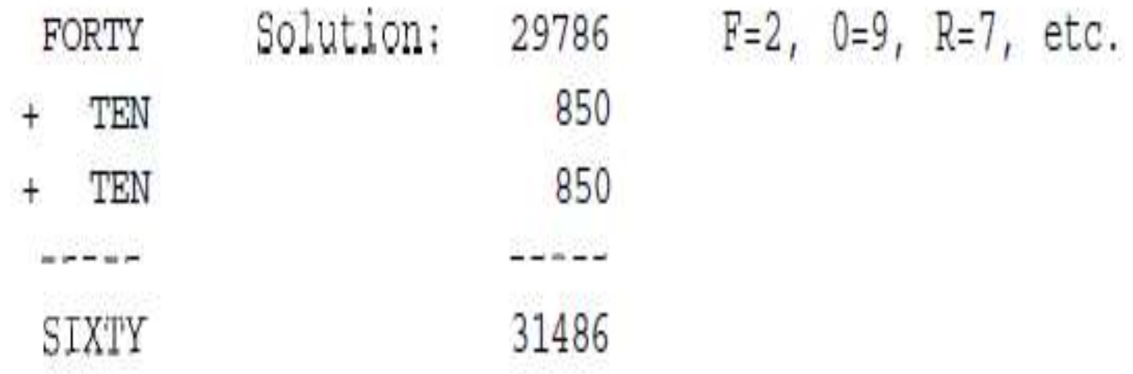
Toy problems

Cryptarithmetic

- Letters stand for digits
- The aim is to find a substitution of digits for letters such that the resulting sum is arithmetically correct.
- Each letter must stand for a different digit.

Problem formulation:

- **States:** a cryptarithmetic puzzle with some letters replaced by digits.
- **Operators:** replace all occurrences of a letter with a digit not already appearing in the puzzle.
- **Goal test:** puzzle contains only digits, and represents a correct sum.
- **Path cost:** zero. All solutions equally valid.



```
 FORTY      Solution:   29786     F=2, O=9, R=7, etc.
+  TEN                    850
+  TEN                    850
 -----                  -----
 SIXTY                  31486
```

Toy problems

The vacuum world

- Assume that the agent knows its location and the locations of all the pieces of dirt, and the suction is still in good working order.
- **States:** one of the eight states
- **Operators:** move left, move right, suck.
- **Goal test:** no dirt left in any square.
- **Path cost:** each action costs 1.

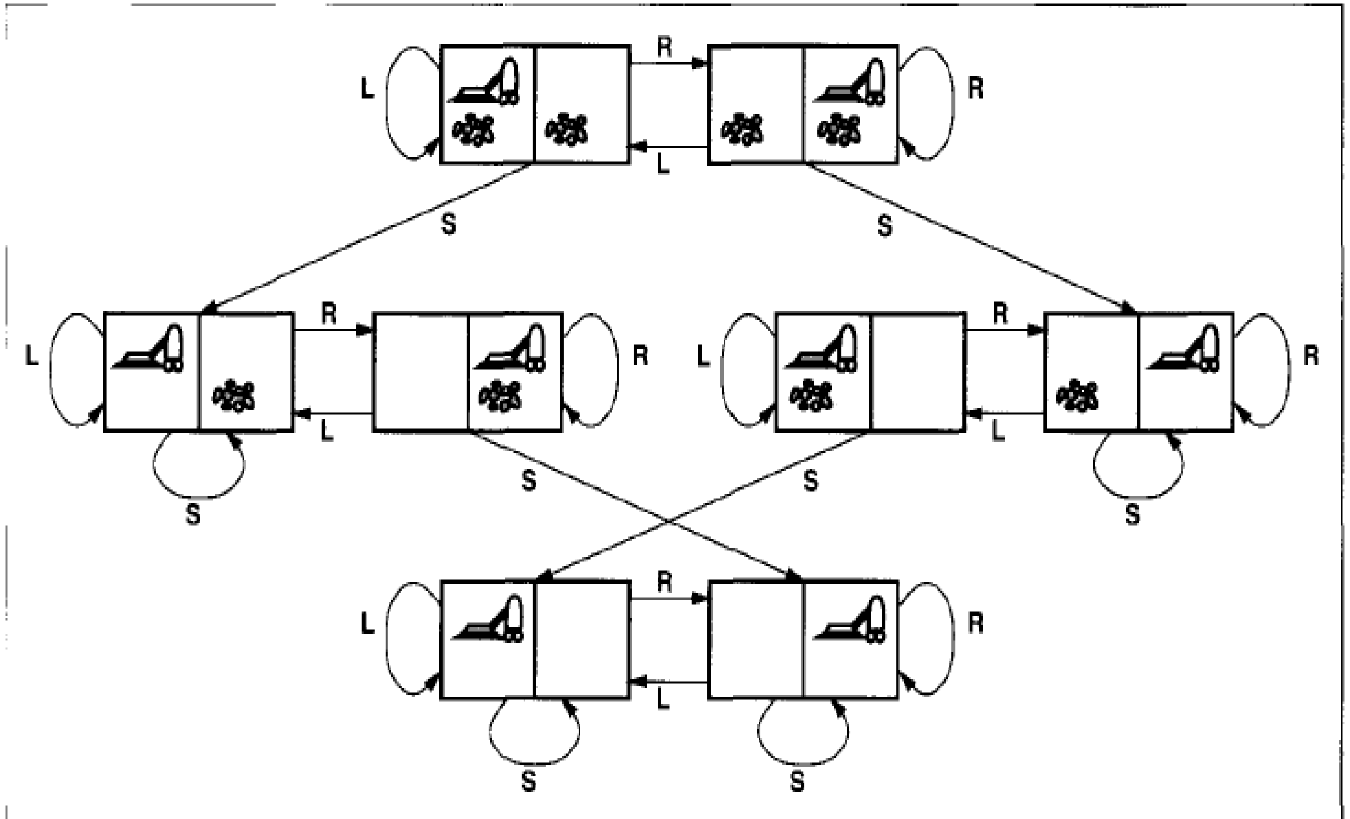


Figure 3.6 Diagram of the simplified vacuum state space. Arcs denote actions. L = move left, R = move right, S = suck.

Real world problems

More difficult and whose solutions people actually care about

Route finding

- routing in

    computer networks

    automated travel advisory systems

    airline travel planning systems.

- the actions in the problem do not have completely known outcomes:

    flights can be late or overbooked

    connections can be missed

    fog or emergency maintenance can cause delays.

- Other real world problems(refer Artificial Intelligence :A Modern Approach by Stuart J. Russell and Peter Norvig page 69)

Touring and travelling salesman problem

VLSI layout

Robot navigation

Assembly sequencing

Problem types and Characteristics

problem types

single-state problem-Agent knows exactly what each of its actions does and it can calculate exactly which state it will be in after any sequence of actions.

multiple-state problem-when the world is not fully accessible, the agent must reason about sets of states that it might get to, rather than single states.

contingency problem-the agent may be in need to now calculate a whole tree of actions, rather than a single action sequence in which each branch of the tree deals with a possible contingency that might arise.

exploration problem-the agent learns a "map" of the environment, which it can then use to solve subsequent problems.

Problem Characteristics:

To choose an appropriate method for a particular Problem:

- Is the problem decomposable?
- Can solution steps be ignored or undone?
- Is the universe predictable?
- Is a good solution absolute or relative?
- Is the solution a state or a path?
- What is the role of knowledge?
- Does the task require human-interaction?

Problem Characteristics

1.Is the problem decomposable?

• Can the problem be broken down to smaller

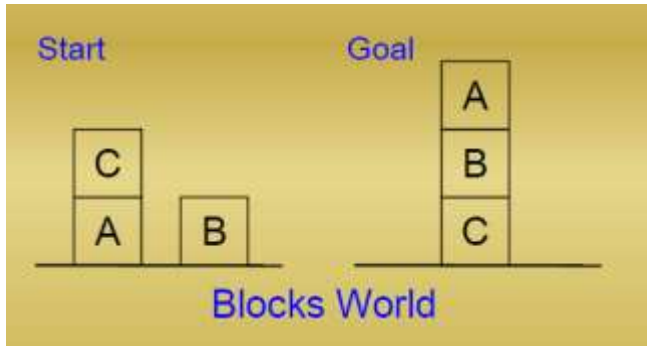
problems to be solved independently?

• Decomposable problem can be solved easily.

Ex 1:- ∫ x2 + 3x + sin2x cos 2x dx

This can be done by breaking it into three smaller problems and solving each by applying specific rules. Adding the results the complete solution is obtained.

Ex2: blocks world problem



Problem Characteristics

2. Can solution steps be ignored or undone?

1. Ignorable problems can be solved using a simple control structure that never backtracks.

Ex:- theorem proving - In which solution steps can be ignored.(comment lines)

2. Recoverable problems can be solved using backtracking.

Ex:- 8 puzzle- In which solution steps can be undone(backtracking and rollback)

3. Irrecoverable problems can be solved by recoverable style methods via planning

Ex:- Chess- In which solution steps can't be undone(Moves cannot be retracted.)

Problem Characteristics

3.Is the universe predictable?

- Certain outcome-8-Puzzle

Every time we make a move, we know exactly what will happen.

- Uncertain outcome-Playing Bridge

We cannot know exactly where all the cards are

or what the other players will do on their turns.

Problem Characteristics

4.Is a good solution absolute or relative?

Eg: Marcus was a man.

Marcus was a Pompeian.

Marcus was born in 40 A.D.

All men are mortal.

All Pompeians died when the volcano erupted in 79 A.D.

No mortal lives longer than 150 years.

It is now 2004 A.D.

Is Marcus alive? (relative)

Different reasoning paths lead to the answer.

It does not matter which path we follow.

The Travelling Salesman Problem (absolute)

We have to try all paths to find the shortest one.

- Any-path problems can be solved using heuristics that suggest good paths to explore.
- For best-path problems, much more exhaustive search will be performed.

Problem Characteristics

5.Is the solution a state or a path?

Finding a consistent intepretation

"The bank president ate a dish of pasta salad with the fork".

– "bank" refers to a financial situation or to a side of a river?

– "dish" or "pasta salad" was eaten?

– Does "pasta salad" contain pasta, as "dog food" does not contain "dog"?

– Which part of the sentence does "with the fork" modify?

- A path-solution problem can be reformulated as a state-solution problem by describing a state as a partial path to a solution.
- The question is whether that is natural or not.

Problem Characteristics

6.What is the role of knowledge?

- Playing Chess

Knowledge is important only to constrain the search for a solution.

- Reading Newspaper

Knowledge is required even to be able to recognize a solution.

Problem Characteristics

7.Does the task require human-interaction?

- **Solitary problem**, in which there is no intermediate communication and no demand for an explanation of the reasoning process.

Eg: mathematical theorems

- **Conversational problem**, in which intermediate communication is to provide either additional assistance to the computer or additional information to the user.

Eg: medical diagnosis

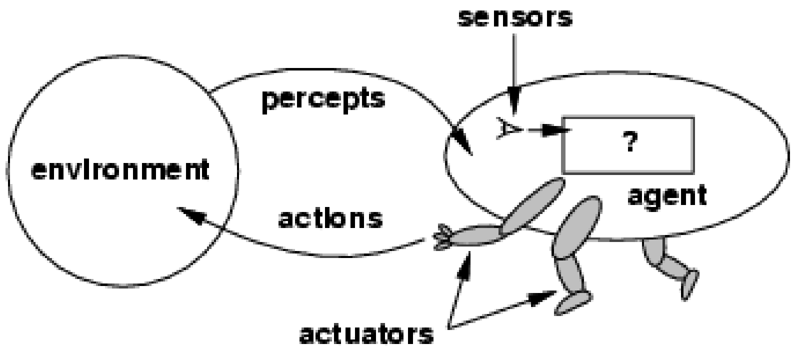
Agents

- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators

- Human agent:
  - eyes, ears, and other organs for sensors;
  - hands, legs, mouth, and other body parts for actuators

- Robotic agent:
  - cameras and infrared range finders for sensors
  - various motors for actuators

Agents and environments

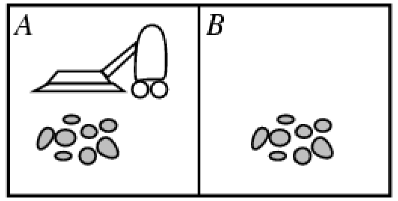


- The agent function maps from percept histories to actions:

$$[f: \mathcal{P}^{\star} \rightarrow \mathcal{A}]$$

- The agent program runs on the physical architecture to produce *f*
- agent = architecture + program

Vacuum-cleaner world



- Percepts: location and contents, e.g., [A,Dirty]
- Actions: *Left*, *Right*, *Suck*, *NoOp*
- *Agent's function → look-up table*
  - *For many agents this is a very large table*

Percept sequence	Action
$[A, Clean]$	$Right$
$[A, Dirty]$	$Suck$
$[B, Clean]$	$Left$
$[B, Dirty]$	$Suck$
$[A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Dirty]$	$Suck$
⋮	⋮

Rational agents

- Rationality
  - Performance measuring success
  - Agents prior knowledge of environment
  - Actions that agent can perform
  - Agent’s percept sequence to date

- Ideal Rational Agent: For each possible percept sequence, an ideal rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

Rationality

- Rational is different from omniscience
  - Percepts may not supply all relevant information
  - E.g., in card game, don’t know cards of others.

- Rational is different from being perfect
  - Rationality maximizes expected outcome while perfection maximizes actual outcome.

Performance measure

- The criteria that determine how successful an agent is.
- One fixed measure is not suitable for all agents
- Performance measures: Safe, fast, legal, comfortable trip, maximize profits
- Example:
  - Agent: Interactive English tutor
  - Performance measure: Maximize student's score on test
  - Environment: Set of students
  - Actuators: Screen display (exercises, suggestions, corrections)
  - Sensors: Keyboard

Flexibility

- Autonomy means “Independence”
- A system is autonomous if its behavior is determined by its own experience
    - An alarm that goes off at a prespecified time is not autonomous
    - An alarm that goes off when smoke is sensed is autonomous
- A system without autonomy lacks flexibility

Autonomy in Agents

- The **autonomy** of an agent is the extent to which its behaviour is determined by its own experience, rather than knowledge of designer.
- Extremes
  - No autonomy – ignores environment/data
  - Complete autonomy – must act randomly/no program
- Example: baby learning to crawl
- Ideal: design agents to have some autonomy
  - Possibly become more autonomous with experience

Intelligent Agent

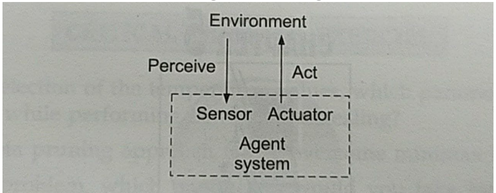


Figure 5.1 Basic structure of an agent.

- Agent - Entity that can perceive the information and act on that information to achieve the desired outcome
- Intelligent Agent - Capable of making decisions and act most logically
- Agent Interacts with environment through sensors and actuators

Intelligent Agent

- Entity that works without assistance, interprets inputs, senses the environment, make choices and acts to achieve the goal

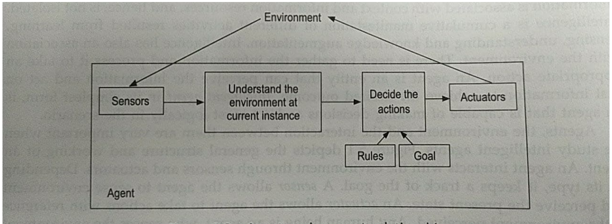


Figure 5.2 Agent-environment relationship.

- Decisions are based on Set of Rules (if-then)
- Intelligent autonomous in Nature, Good Observant to the environment

Intelligent Agent - Cont'd

- Percept Sequence - Complete history of everything the agent has ever captured
- Agent's choice of an action depends on percept sequence
- Agent Function maps actions and percept sequences
- Agent Function is generally implemented by an internal program called agent program
- Mapping of Agent Function is

$$f : P \rightarrow A$$

- Agent consists of architecture and program

$$\text{Sensors} \rightarrow \text{Agent Programs} \rightarrow \text{Actuators}$$

- Operation of agent is dependent on the function

Rationality and Rational Agent

- Rational Agent is an agent that behaves logically and does the right things
- Rationality is a normative concept that stands for acting based on logical reasoning
- Rational Agent is expected to select an action that would maximize the performance with respect to the percept sequence and Built-in knowledge

Rationality and Rational Agent - cont'd

Parameters for consideration:

- Priorities and preferences of the agent
- Environment information available in the agents
- Possible actions for agents
- Estimated or actual benefits and chances of success of the actions

Agents in AI

- Performance Measures is used to track the performance of an agent
- The selected action by the agents changes the state of environment
- If the sequence of selected actions is desirable, agent did very well
- Performance measure based on what one actually wants from environment

Rationality and Performance

- Rationality is with knowledge available with the agent
- Knowledge is derived from the percept sequence to date
- Learning agent which is part of rational agent that can succeed in variety of environments
- Rational agent should possess the following properties
  - Ability to gather information
  - Ability to learn from the experience
  - Perform knowledge augmentation
  - Autonomy

Flexibility and Intelligent Agent

- Intelligence demands flexibility and agents do need that
- Agents need to be
  - Responsive (Timely fashion reaction)
  - Pro-active (Opportunistic and goal-directed behaviour)
  - Social (Interaction with other agents)
  - Mobility (to accumulate the knowledge)
  - Veracity (Truthful)
  - Benevolence (Avoiding conflict)
  - Rationality (Act to maximize the expected performance)
  - Learning (Increase in learning increases the performances)
- For designing an intelligent agent, PEAS (P - Performance, E - Environment, A - Agent's actuators, S - Sensors) are necessary

Types of Agents

- Types based on Complexity and functionality (Expected Intelligence) of agent
- Complexity of the environment contributes contributes to complexity of the agent architecture
- Types of Agents are,
  - Table-driven agents
  - Simple Reflex Agents
  - Model-based reflex Agents
  - Goal-based Agents
  - Utility-based Agents

Table-driven Agents

- Agents are based on simple table which entries determine the action (Percept to Percept Sequence)
- **Drawbacks:**
  - Size of the Table
  - Time to build the table
  - No Autonomy
  - Time required for learning

Simple Reflex Agents

- Agents accountable only to the current percept
- Action is decided based on rules (Condition)
- Agents are limited with intelligence
- Can not handle the complex scenario
- Simple Reflex Agents may stuck in infinite loops incase of unavailability of some sensors
- Use of randomise simple reflex agents can solve problem (to some extent)
- Randomization may impact rational behaviour

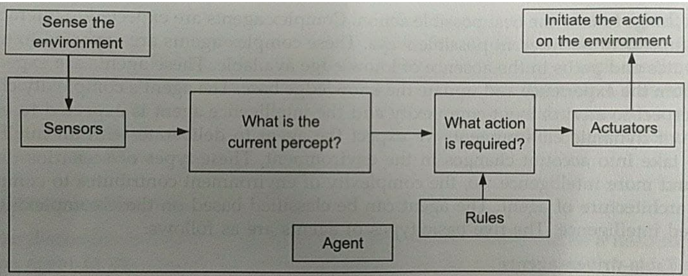
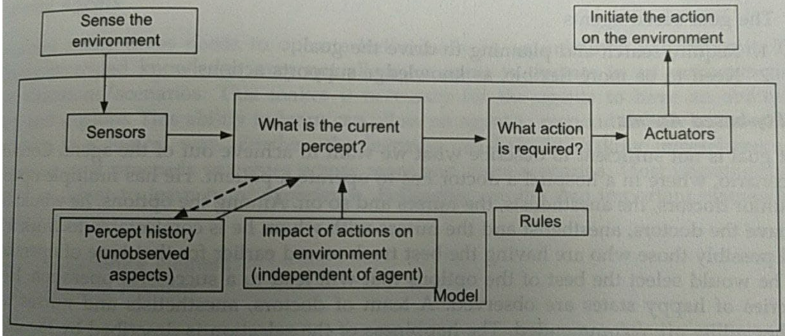


Figure 5.3 Simple reflex agents.

Model-based Reflex Agents

- Percept history is maintained (Using internal states)
- Agents reflect some of the unobserved aspects of the current state
- Incorporating things in determining the unobserved aspects is called model
- Model keeps track the past sequence of percept to determine the unseen part and impact of actions on environment
- Modle-based intelligent agents have wide applications compared to simple reflex agents



Sense the environment
Initiate the action on the environment
Sensors
What is the current percept?
What action is required?
Actuators
Rules
Percept history (unobserved aspects)
Impact of actions on environment (independent of agent)
Model

Figure 5.4 Model-based reflex agents.

Goal-based Agents

- Goal of Decision-Making is considered
- Goal based action is very complicated and need a sequence of actions
- Decision-Making checks the impact of actions with reference to the goal
- Less efficient but more flexible
- More time is required for reasoning process
- Resemblance of Model-Based agents
- Goal based agents require
  - Require search and planning to drive the goal
  - Need to be more flexible, as knowledge supports action

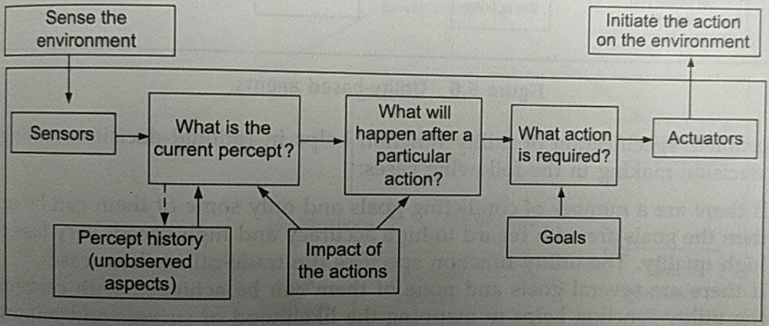

Sense the environment
Initiate the action on the environment
Sensors
What is the current percept?
What will happen after a particular action?
What action is required?
Actuators
Percept history (unobserved aspects)
Impact of the actions
Goals

Figure 5.5 Goal-based agents.

Utility-Based Agents

- Selection is described by the utility function
- Utility Function is most often a real number that maps to degree of happiness
- Complete specification of utility function helps in rational decision-making
- Utility-based Agents can help in decision-making in following cases
  - Only certain goals could be achieved in conflicting goals with respect to some performance measure
  - Utility function maps to likelihood of success and importance of goal in order to take decision when goals are uncertain
  - Problems related to route selection and modification

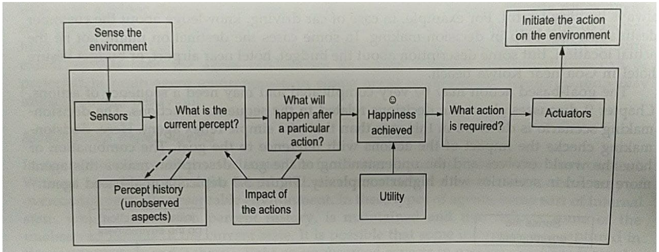

Sense the environment
Sensors
What is the current percept?
What will happen after a particular action?
☺ Happiness achieved
What action is required?
Actuators
Initiate the action on the environment
Percept history (unobserved aspects)
Impact of the actions
Utility

Figure 5.6 Utility-based agents.

Utility-based agents are useful when:

- Not only goals, but means are also important
- Some series of actions are safer, quicker and reliable
- Happy and Unhappy states are required to be distinguished

Learning Agent

- Learning agent responds new or unknown scenarios in a logical way
- Agents learn from experiences
- Two important elements in leanring agents
  - Performance Agents - selection of external actions
  - Learning Agents - Responsible for making improvements
- Feedback is provided for learning which makes improvements in name of penalty or reward
- Task Generator or Problem Generator suggests actions that generate new experiences
- Learning is the process of modification of each component of agent to bring to reach the expected

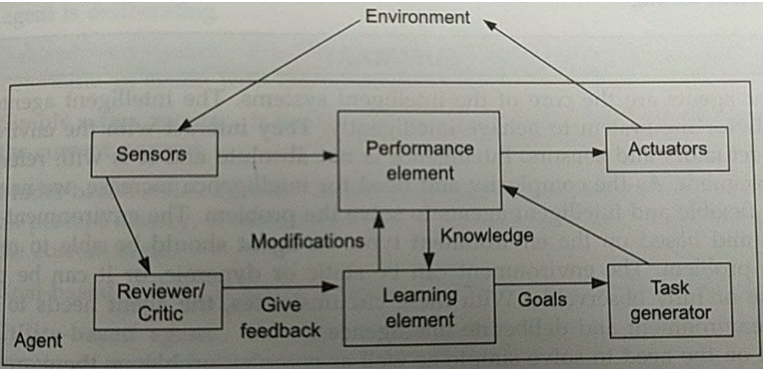


Figure 5.7 Learning agent.

Constraint Satisfaction Problems (CSP)

- Search Space is constrained by a set of conditions and dependencies
- Class of problems where the search space is constrained called as CSP
- For example, Time-table scheduling problem for lecturers.
- Constraint,
  - Two Lecturers can not be assigned to same class at the same time
- To solve CSP, the problem is to be decomposed and analyse the structure
- Constraints are typically mathematical or logical relationships

CSP

- Any problem in the world can be mathematically represented as CSP
- Hypothetical problem does not have constraints
- Constraint restricts movement, arrangement, possibilities and solutions
- Example: if only concurrency notes of 2,5,10 reupees are available and we need to give certain amount of money, say Rs. 111 to a salesman and the total number of notes should be between 40 and 50.
- Expressed as, Let n be number of notes,

$$40 < n < 50$$

and

$$c_1X_1 + c_2X_2 + c_3X_3 = 111$$

CSP

- Types of Constraints
  - Unary Constraints - Single variable
  - Binary Constraints - Two Variables
  - Higher Order Constraints - More than two variables
- CSP can be represented as Search Problem
- Initial state is empty assignment, while successor function is a non-conflicting value assigned to an unassigned variables
- Goal test checks whether the current assignment is complete and path cost is the cost for the path to reach the goal state
- CSP Solutions leads to the final and complete assignment with no exception

Crypt-arithmetic Puzzles

- Cryptarithmetic puzzles are also represented as CSP
- Example:
  MIKE +
  JACK =
  JOHN
- Replace every letter in puzzle with single number (number should not be repeated for two different alphabets)
- The domain is { 0,1, ... , 9 }
- Often treated as the ten-variable constraint problem where the constraints are:
  - All the variables should have a different value
  - The sum must work out

Cryptarithmetic Puzzles

- $M * 1000 + I * 100 + K * 10 + E + J * 1000 + A * 100 + C * 10 + K = J * 1000 + O * 100 + H * 10 + N$
- Constraint Domain is represented by Five-tuple and represented by,
$$D = \{var, f, O, dv, rg\}$$
- *Var* stands for set variables, f is set functions, O stands for the set of legitimate operators to be used, *dv* is domain variable and *rg* is range of function in the constraint
- Constraint without conjunction is referred as Primitive constraint (for Eg., $x < 9$)
- Constraint with conjunction is called as non-primitive constraint or a generic constraint (For Eg., $x < 9$ and $x > 2$)

Room Coloring Problem

- Let K for Kitchen, D for Dining Room, H is for Hall, B_2 and B_3 are bedrooms 2 and 3, MB_1 is master bedroom, SR is the store Room, GR is Guest Room and Lib is Library
- Constraints
  - All bedrooms should not be colored red, only one can
  - No two adjacent rooms can have the same color
  - The colors available are red, blue, green and violet
  - Kitchen should not be colored green
  - Recommended to color the kitchen as blue
  - Dining room should not have violet color

Room Coloring Problem

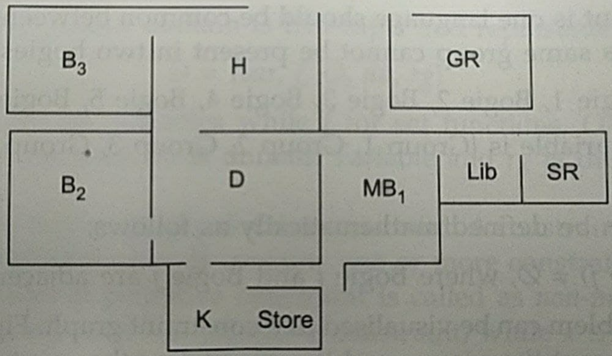


Figure 6.3 Home colouring problem.

Representation as a search problem

- Lines - Adjacency rooms
- Dotted Lines - No Adjacency

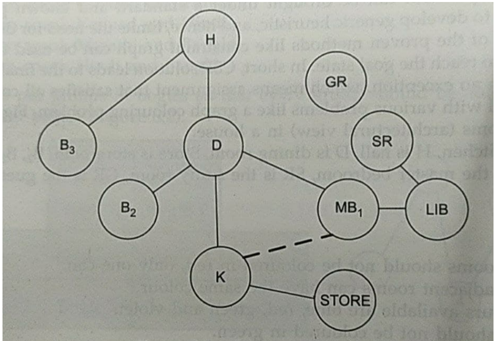
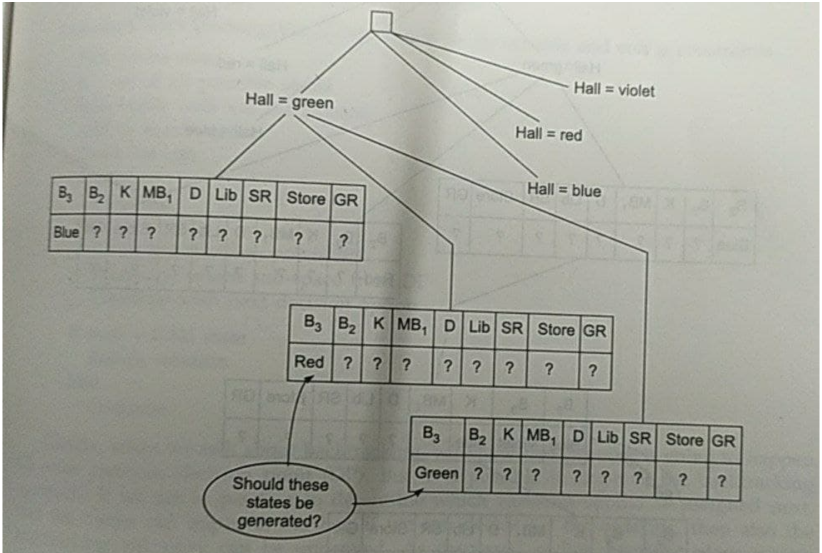


Figure 6.4 Constraint graph.

Room Coloring Problem

- Soft Constraints are that they are cost-oriented or preferred choice
- All paths in the Search tree can not be accepted because of the violation in constraints



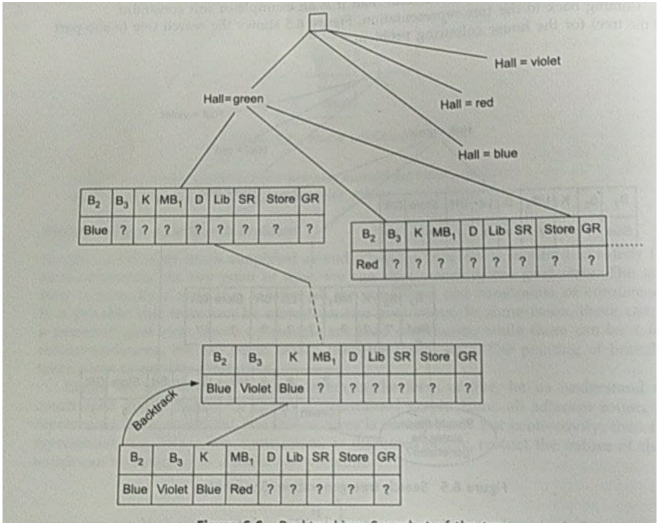
Hall = green
Hall = violet
Hall = red
Hall = blue
B3 | B2 | K | MB1 | D | Lib | SR | Store | GR
Blue | ? | ? | ? | ? | ? | ? | ? | ?
B3 | B2 | K | MB1 | D | Lib | SR | Store | GR
Red | ? | ? | ? | ? | ? | ? | ? | ?
B3 | B2 | K | MB1 | D | Lib | SR | Store | GR
Green | ? | ? | ? | ? | ? | ? | ? | ?
Should these states be generated?

Figure 6.5 Search tree generation: Snapshot.

Backtracking Search for CSP

- Assignment of value to any additional variable within constraint can generate a legal state (Leads to successor state in search tree)
- Nodes in a branch backtracks when there is no options are available

Backtracking Search for Room Coloring



Hall = violet
Hall = red
Hall=green
Hall = blue
B2 B3 K MB1 D Lib SR Store GR
Blue ? ? ? ? ? ? ? ?
B2 B3 K MB1 D Lib SR Store GR
Red ? ? ? ? ? ? ? ?
B2 B3 K MB1 D Lib SR Store GR
Blue Violet Blue ? ? ? ? ? ?
Backtrack
B2 B3 K MB1 D Lib SR Store GR
Blue Violet Blue Red ? ? ? ? ?

Figure 6.6 Backtracking: Snapshot of the tree.

Algorithm for Backtracking

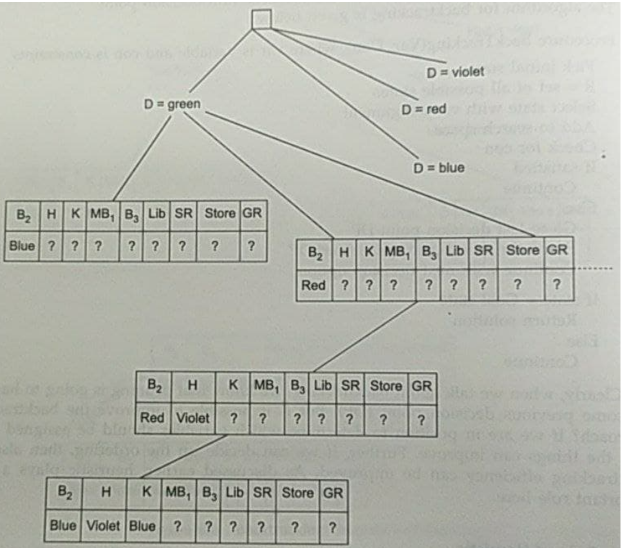
```
Pick initial state
R = set of all possible states
Select state with var assignment
Add to search space
check for con
If Satisfied
    Continue
Else
    Go to last Decision Point (DP)
    Prune the search sub-space from DP
    Continue with next decision option
If state = Goal State
    Return Solution
Else
    Continue
```

Backtracking Search for CSP

- Backtracking allows to go to the previous decision-making node to eliminate the invalid search space with respect to constraints
- Heuristics plays a very important role here
- If we are in position to determine which variables should be assigned next, then backtracking can be improved
- Heuristics help in deciding the initial state as well as subsequent selected states
- Selection of a variable with minimum number of possible values can help in simplifying the search
- This is called as Minimum Remaining Values Heuristic (MRV) or Most Constraint Variable Heuristic
- Restricts the most search which ends up in same variable (which would make the backtracking ineffective)

Heuristics

- MRV cannot have hold on initial selection process
- Node with maximum constraint is selected over other unassigned variables - Degree Heuristics
- By degree heuristics, branching factor can not be reduced
- Selection of variables are considered not the values for it, so the order in which the values of particular variable can be arranged is tackled by least constraining value heuristic



B_2	H	K	MB_1	B_3	Lib	SR	Store	GR
Blue	?	?	?	?	?	?	?	?

B_2	H	K	MB_1	B_3	Lib	SR	Store	GR
Red	?	?	?	?	?	?	?	?

B_2	H	K	MB_1	B_3	Lib	SR	Store	GR
Red	Violet	?	?	?	?	?	?	?

B_2	H	K	MB_1	B_3	Lib	SR	Store	GR
Blue	Violet	Blue	?	?	?	?	?	?

Figure 6.7 Maximum constrained variable selection.

Forward Checking

- To understand the forward checking, we shall see 4 Queens problem
- If an arrangement on the board of a queen x, hampers the position of $queens_{x+1}$, then this forward check ensures that the queen x should not be placed at the selected position and a new position is to be looked upon

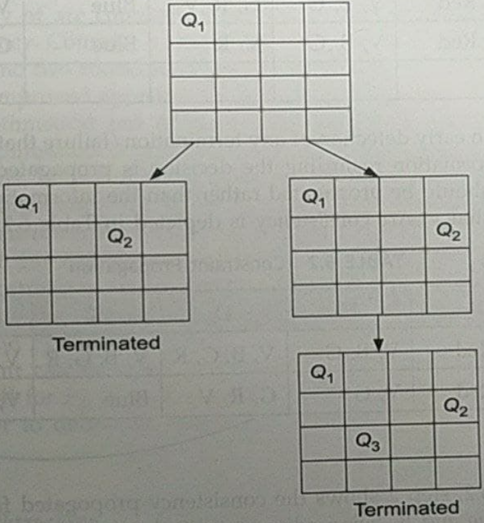


Figure 6.8 Forward checking for four-queens.

Foward Checking

- Q_1 and Q_2 are placed in row 1 nad 2 in the left sub-tree, so, search is halted, since No positions are left for Q_3 and Q_4
- Forward Checking keeps track of the next moves that are available for the unassigned variables
- The search will be terminated when there is no legal move available for the unassigned variables

Forward Checking

- For Room Coloring problem, Considering all the constraints the mapping can be done in following ways;
  - At first, B_2 is selected with Red (R). Accordingly, R is deleted from the adjacent nodes
  - Kitchen is assigned with Blue (B). So, B is deleted form the adjacent Nodes
  - Furthermore, as MB_1 is selected green, no color is left for D.

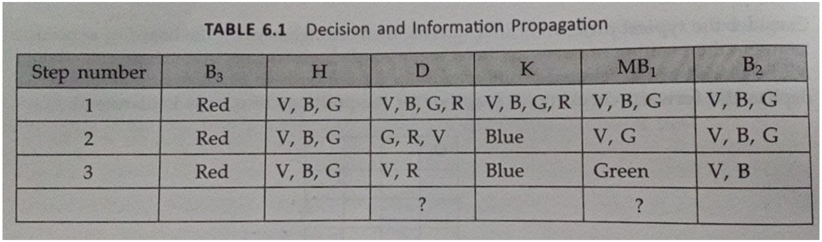


TABLE 6.1 Decision and Information Propagation

Step number	B_3	H	D	K	MB_1	B_2
1	Red	V, B, G	V, B, G, R	V, B, G, R	V, B, G	V, B, G
2	Red	V, B, G	G, R, V	Blue	V, G	V, B, G
3	Red	V, B, G	V, R	Blue	Green	V, B
			?		?	

Constraint Propagation

- There is no early detection of any termination / failure that would possible occur even though the information regarding the decision is propagated
- Constraint should be propagated rather than the information

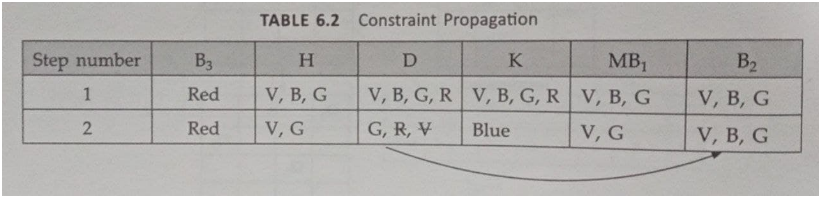


TABLE 6.2 Constraint Propagation

Step number	B_3	H	D	K	MB_1	B_2
1	Red	V, B, G	V, B, G, R	V, B, G, R	V, B, G	V, B, G
2	Red	V, G	G, ~~R~~, ~~V~~	Blue	V, G	V, B, G

Constraint Propagation

- Step 2 shows the consistency propagated from D to B_2
- Since D is can have only G value and B_2 being adjacent to it, the arc is drawn
- It is mapped as $D \rightarrow B_2$ or Mathematically,

$$A \rightarrow B \text{ is consistent} \leftrightarrow \forall \text{ legal value } a \in A, \exists$$

$$\text{non-conflicting value b} \in B$$

- Failure detection can take place at early stage

Constraint Propagation

- Algorithm for arc assignment is:
  - Let C be the variable which is being assigned at a given instance
  - X will have some value from D{} where D is domain
  - For each and every assigned variable, that is adjacent to X, Say X'
    1. Perform forward check (remove values from domain D that conflict the decision of the current assignment)
    2. For every other variable X'' that are adjacent or connected to X' ;

    i Remove the values from D from X'' that can't be taken as further unassigned variables

    ii Repeat step 2, till no more values can be removed or discarded
  - Inconsistency is considered and constraints are propagated in Step (2)

Intelligent Backtracking

- Conflict set is maintained using forward checking and maintained
- Considering the 4 Queens problem, Conflict needs to be detected by the user of conflict set so that a backtrack can occur
- Backtracking with respect to the conflict set is called as conflict-directed backjumping
- Backjumping approach can't actually restrict the earlier committed mistakes in some other branches

References

- *Parag Kulkarni, Prachi Joshi, Artificial Intelligence –Building Intelliegent Systems, 1st ed., PHI learning,2015*
- *Stuart J. Russell, Peter Norwig , Artificial Intelligence –A Modern approach, 3rd Pearson Education, 2016*
- Kevin Night and Elaine Rich, Nair B., “Artificial Intelligence (SIE)”, Mc Graw Hill- 2008.