Unit 1 List of Topics

- **Introduction to AI** -AI techniques
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

What is Artificial Intelligence?

- A.I. is the study of how to make computers do things at which, at the moment, people are better.

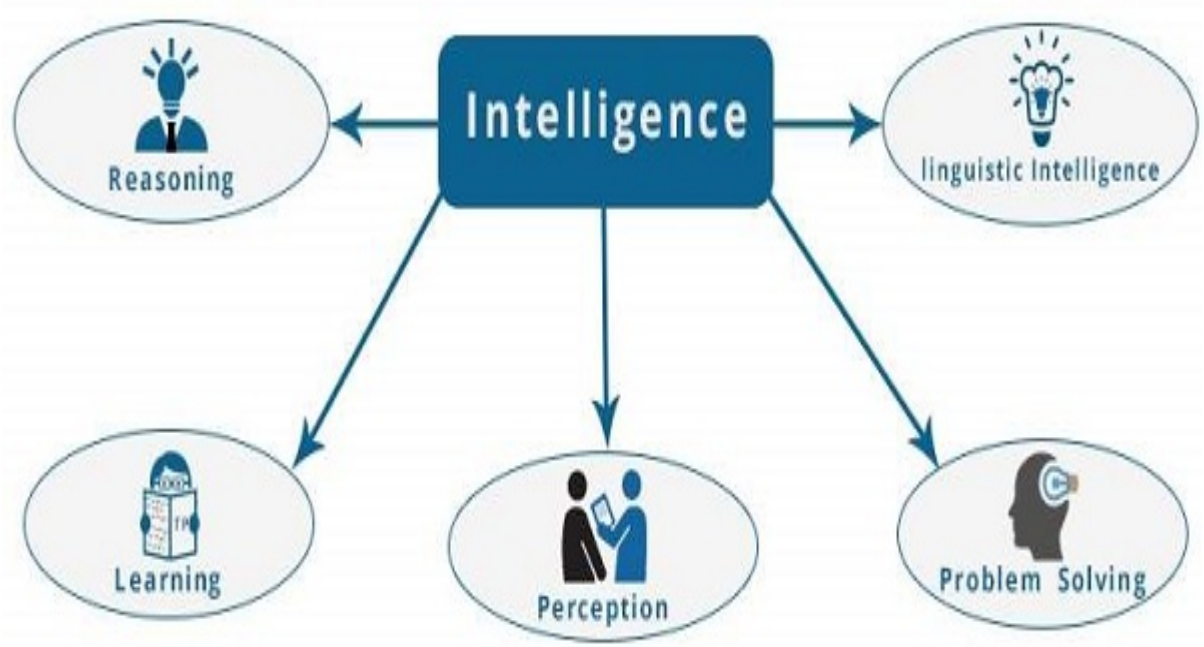
Intelligence vs Artificial Intelligence

- Intelligence is a property/ability attributed to people, such as to know, to think, to talk, to learn, to understand.

Intelligence = Knowledge + ability to perceive, feel, comprehend,process, communicate, judge, learn.

- Artificial Intelligence is an interdisciplinary field aiming at developing techniques and tools for solving problems that people are good at.

Intelligence



Definitions of AI

Existing definitions advocate everything from replicating human intelligence to simply solving knowledge-intensive tasks. Examples:

- “Artificial Intelligence is the design, study and construction of computer programs that behave intelligently.” --Tom Dean.
- “Artificial Intelligence is the enterprise of constructing a physical symbol system that can reliably pass the Turing test.” --Matt Ginsberg.

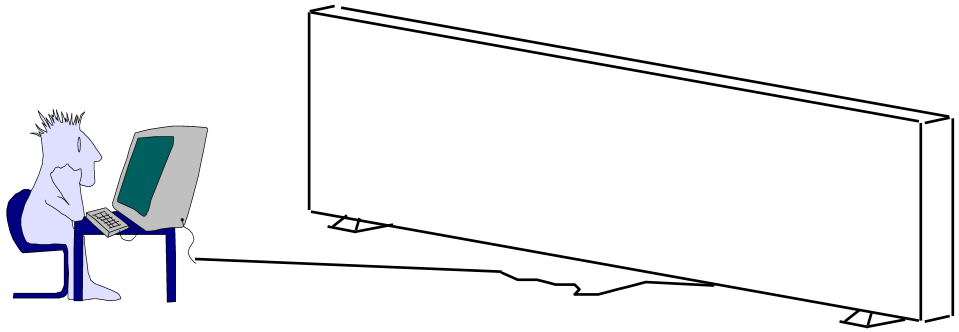
What is Artificial Intelligence ?

THOUGHT	**Systems that think like humans**	**Systems that think rationally**
BEHAVIOUR	**Systems that act like humans**	**Systems that act rationally**
	HUMAN	**RATIONAL**

Systems that act like humans: Turing Test

- "The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil)
- "The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight)

Systems that act like humans



?

- You enter a room which has a computer terminal. You have a fixed period of time to type what you want into the terminal, and study the replies. At the other end of the line is either a human being or a computer system.
- If it is a computer system, and at the end of the period you cannot reliably determine whether it is a system or a human, then the system is deemed to be intelligent.

Systems that act like humans

- The Turing Test approach
  - a human questioner cannot tell if
    - there is a computer or a human answering his question, via teletype (remote communication)
  - The computer must behave intelligently
- Intelligent behavior
  - to achieve human-level performance in all cognitive tasks

Systems that act like humans

- These cognitive tasks include:
  - *Natural language processing*
    - for communication with human
  - *Knowledge representation*
    - to store information effectively & efficiently
  - *Automated reasoning*
    - to retrieve & answer questions using the stored information
  - *Machine learning*
    - to adapt to new circumstances

The total Turing Test

- Includes two more issues:
  - *Computer vision*
    - to perceive objects (seeing)
  - *Robotics*
    - to move objects (acting)

What is Artificial Intelligence ?

	HUMAN	RATIONAL
THOUGHT	Systems that think like humans	Systems that think rationally
BEHAVIOUR	Systems that act like humans	Systems that act rationally

Systems that think like humans: cognitive modeling

- Humans as observed from 'inside'
- How do we know how humans think?
  - Introspection vs. psychological experiments
- Cognitive Science
- "The exciting new effort to make computers think … machines with *minds* in the full and literal sense" (Haugeland)
- "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning …" (Bellman)

What is Artificial Intelligence ?

THOUGHT	Systems that think like humans	Systems that think rationally
BEHAVIOUR	Systems that act like humans	Systems that act rationally
	HUMAN	RATIONAL

Systems that think ‘rationally’ "laws of thought"

- Humans are not always ‘rational’
- Rational - defined in terms of logic?
- Logic can’t express everything (e.g. uncertainty)
- Logical approach is often not feasible in terms of computation time (needs ‘guidance’)
- “The study of mental facilities through the use of computational models” (Charniak and McDermott)
- “The study of the computations that make it possible to perceive, reason, and act” (Winston)

What is Artificial Intelligence ?

THOUGHT	**Systems that think like humans**	**Systems that think rationally**
BEHAVIOUR	**Systems that act like humans**	**Systems that act rationally**
	HUMAN	**RATIONAL**

Systems that act rationally: "Rational agent"

- Rational behavior: doing the right thing
- The right thing: that which is expected to maximize goal achievement, given the available information
- Giving answers to questions is 'acting'.
- I don't care whether a system:
  - replicates human thought processes
  - makes the same decisions as humans
  - uses purely logical reasoning

Systems that act rationally

- Logic □ only *part* of a rational agent, not *all* of rationality
  - Sometimes logic cannot reason a correct conclusion
  - At that time, some *specific (in domain) human knowledge* or information is used
- Thus, it covers more generally different situations of problems
  - Compensate the incorrectly reasoned conclusion

Systems that act rationally

- Study AI as rational agent –

2 advantages:

  - It is more general than using logic only
    - Because: LOGIC + Domain knowledge
  - It allows extension of the approach with more scientific methodologies

Rational agents

- An agent is an entity that perceives and acts
- This course is about designing rational agents
- Abstractly, an agent is a function from percept histories to actions:

$$[f: P^* \rightarrow A]$$

- For any given class of environments and tasks, we seek the agent (or class of agents) with the best performance
- Caveat: computational limitations make perfect rationality unachievable
  - design best program for given machine resources

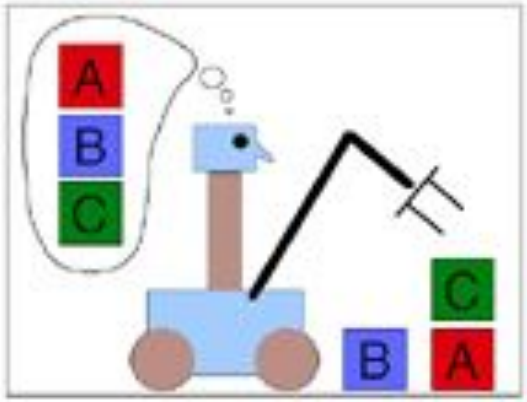
- Artificial
  - Produced by human art or effort, rather than originating naturally.
- Intelligence
- is the ability to acquire knowledge and use it" [Pigford and Baur]
- **So AI was defined as:**
  - AI is the study of ideas that enable computers to be intelligent.
  - AI is the part of computer science concerned with design of computer systems that exhibit human intelligence(From the Concise Oxford Dictionary)

From the above two definitions, we can see that AI has two major roles:

- Study the intelligent part concerned with humans.
- Represent those actions using computers.

Goals of AI

- To make computers more useful by letting them take over dangerous or tedious tasks from human
- Understand principles of human intelligence



The Foundation of AI

- ***Philosophy***
  - At that time, the study of human intelligence began with no formal expression
  - Initiate the idea of mind as a machine and its internal operations

The Foundation of AI

- Mathematics formalizes the three main area of AI: *computation*, *logic*, and *probability*
  - Computation leads to analysis of the problems that can be computed
    - *complexity theory*
  - Probability contributes the *"degree of belief"* to handle *uncertainty* in AI
  - *Decision theory* combines *probability theory* and *utility theory* (bias)

The Foundation of AI

- Psychology
  - How do humans think and act?
  - The study of human reasoning and acting
  - Provides reasoning models for AI
  - Strengthen the ideas
    - humans and other animals can be considered as information processing machines

The Foundation of AI

- Computer Engineering
  - How to build an efficient computer?
  - Provides the artifact that makes AI application possible
  - The power of computer makes computation of large and difficult problems more easily
  - AI has also contributed its own work to computer science, including: time-sharing, the linked list data type, OOP, etc.

The Foundation of AI

- **Control theory and Cybernetics**
  - How can artifacts operate under their own control?
  - The artifacts adjust their actions
    - To do better for the environment over time
    - Based on an objective function and feedback from the environment
  - Not limited only to linear systems but also other problems
    - as language, vision, and planning, etc.

The Foundation of AI

- Linguistics
  - For understanding natural languages
    - different approaches has been adopted from the linguistic work
  - Formal languages
  - Syntactic and semantic analysis
  - Knowledge representation

The main topics in AI

Artificial intelligence can be considered under a number of headings:

- Search (includes Game Playing).
- Representing Knowledge and Reasoning with it.
- Planning.
- Learning.
- Natural language processing.
- Expert Systems.
- Interacting with the Environment
  (e.g. Vision, Speech recognition, Robotics)

Some 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.

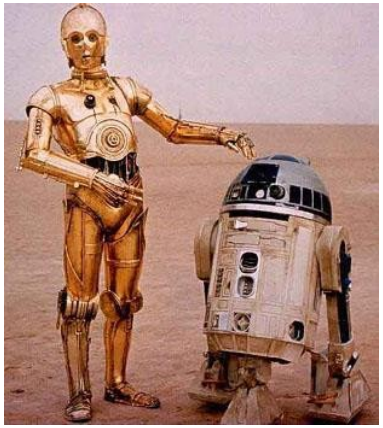
AI – Social Companion

Nadine the Robot Is Your New Social Companion

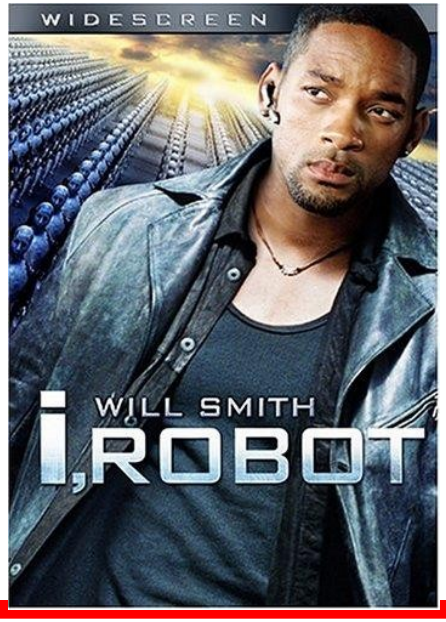
After the Coronavirus epidemic there has been an increased demand for robots for various activities like serving food in hospitals...

.....even as companions

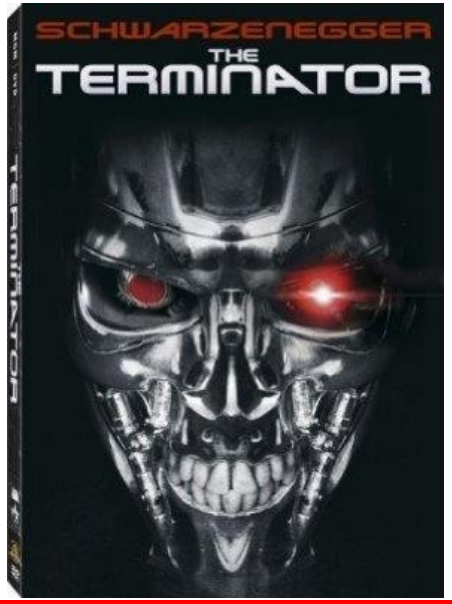
AI in Movies




2001:
a space odyssey


WIDESCREEN
WILL SMITH
i,ROBOT


BATTLESTAR
GALACTICA


SCHWARZENEGGER
THE
TERMINATOR

AI Applications

AI Defined

- Textbook definition:

  – AI may be defined as the branch of computer science that is concerned with the automation of intelligent behavior

Applied Areas of AI

- Heuristic Search
- Computer Vision
- Adversarial Search (Games)
- Fuzzy Logic
- Natural Language Processing
- Knowledge Representation
- Planning
- Learning

Examples

- Playing chess
- Driving on the highway
- Mowing the lawn
- Answering questions
- Recognizing speech
- Diagnosing diseases
- Translating languages
- Data mining

Unit 1 List of Topics

- Introduction to AI- **AI techniques**
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

AI Techniques

- AI technique is a method that achieves knowledge. The main AI techniques are:
  - Search
  - Use of knowledge
  - Abstraction

1. Search:-

- Search provides a way of solving problems for which no more direct approach is available as well as a framework into which any direct techniques that are available can be embedded. A search program finds a solutions for a problem by trying various sequences of actions or operators until a solution is found.

Advantages

- It is the best way so far as no better way has been found to solve the problems.
- To solve a problem using search, it is only necessary to code the operator that can be used; the search will find the sequence of actions that will provide the desired results.

Disadvantages

- Most problems have search spaces so large that it is impossible to search for the whole space.

AI Techniques

2. Use of knowledge:-

- The use of knowledge provides a way of solving complicated problems by manipulating the structures of the objects that are concerned.
- The way in which knowledge can be represented for usage in AI techniques:
- AI technique is a method that achieves knowledge that should be represented in such a way that:-
- Knowledge captures generalization. This meaning grouping situations that share important properties rather than representing each situation separately with such an arrangement of knowledge, an unreasonable amount of memory, and updating will no longer be required. Anything without this property is called data rather than knowledge.
- It should be represented in such a way that it can be understood by the people who must prepare it. For many programs, the size of the data can be achieved automatically by taking a reading from a number of instruments, but in many AI areas, most of the knowledge a program has must be provided by people in terms that they understand it.
- It could easily be adjusted to correct errors end to demonstrate changes in the world.
- It can be used to overcome its own through volume by helping to restrict the range of possibilities that must usually be considered or discussed.
- It could be used in different situations even though it may not entirely be complete.

3. Abstraction:-

- Abstraction finds a way of separating important features and notifications from the unimportant ones that would otherwise confuse any process.

AI Techniques

AI technique is a method that exploits knowledge that should be represented in such a way that:

1. The knowledge captures generalizations
2. It can be understood by people who must provide it.
3. It can easily be modified to correct errors
4. It can be used in a great many situations even if it is not totally accurate or complete
5. It can be used to help overcome its own sheer bulk by helping to narrow the range of possibilities that must usually be considered.

AI Techniques

- An AI technique is a method that exploits knowledge that is represented so that: The knowledge captures generalizations that share properties, are grouped together, rather than being allowed separate representation.
- It can be understood by people who must provide it—even though for many programs bulk of the data comes automatically from readings.
- In many AI domains, how the people understand the same people must supply the knowledge to a program.
- It can be easily modified to correct errors and reflect changes in real conditions.
- It can be widely used even if it is incomplete or inaccurate.
- It can be used to help overcome its own sheer bulk by helping to narrow the 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.

Unit 1 List of Topics

- Introduction to AI-AI techniques
- **Problem solving with AI**
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

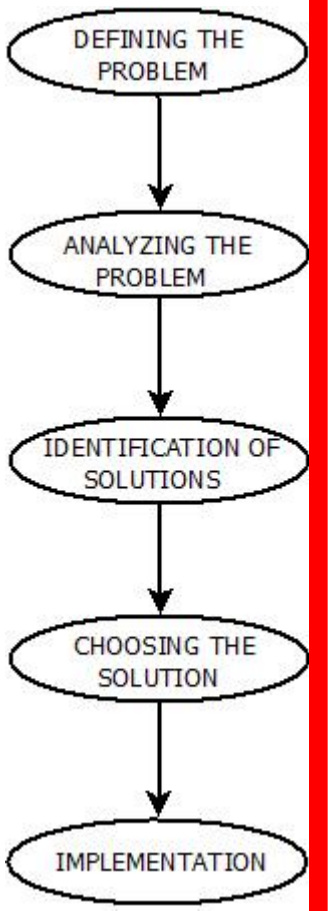
Problem Solving with AI

Problem Solving In AI : Introduction

- Problem Solving in games such as “Sudoku” can be an example. It can be done by building an artificially intelligent system to solve that particular problem. To do this, one needs to define the problem statements first and then generating the solution by keeping the conditions in mind.
- Some of the most popularly used problem solving with the help of artificial intelligence are:
  - Chess.
  - Travelling Salesman Problem.
  - Tower of Hanoi Problem.
  - Water-Jug Problem.
  - N-Queen Problem.
- Problem Searching
- In general, searching refers to as finding information one needs.
- Searching is the most commonly used technique of problem solving in artificial intelligence.
- The searching algorithm helps us to search for solution of particular problem.

Problem Solving In AI : Introduction

- **Problem**
- **Problems are the issues which comes across any system. A solution is needed to solve that particular problem.**
- **Steps : Solve Problem Using Artificial Intelligence**
- **The process of solving a problem consists of five steps. These are:**



Problem Solving In AI : Introduction

- **Defining The Problem: The definition of the problem must be included precisely. It should contain the possible initial as well as final situations which should result in acceptable solution.**
- **Analyzing The Problem: Analyzing the problem and its requirement must be done as few features can have immense impact on the resulting solution.**
- **Identification Of Solutions: This phase generates reasonable amount of solutions to the given problem in a particular range.**
- **Choosing a Solution: From all the identified solutions, the best solution is chosen basis on the results produced by respective solutions.**
- **Implementation: After choosing the best solution, its implementation is done.**

Broad Categorisation of AI problems

- Structured Problem
  - Well structed – Yield a right answer
  - Ill structed – Do not yield a particular answer
- Unstructured Problem
  - Very hard to formulate the problem
  - Ambiguous in nature
- Linear Problem
  - Have clear solution
  - All kind of classification problems
- Non linear Problem
  - Relationships between input and output is non linear
  - Further decision can't be taken like in linear problem

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- **AI Models**, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents

- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles

- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic

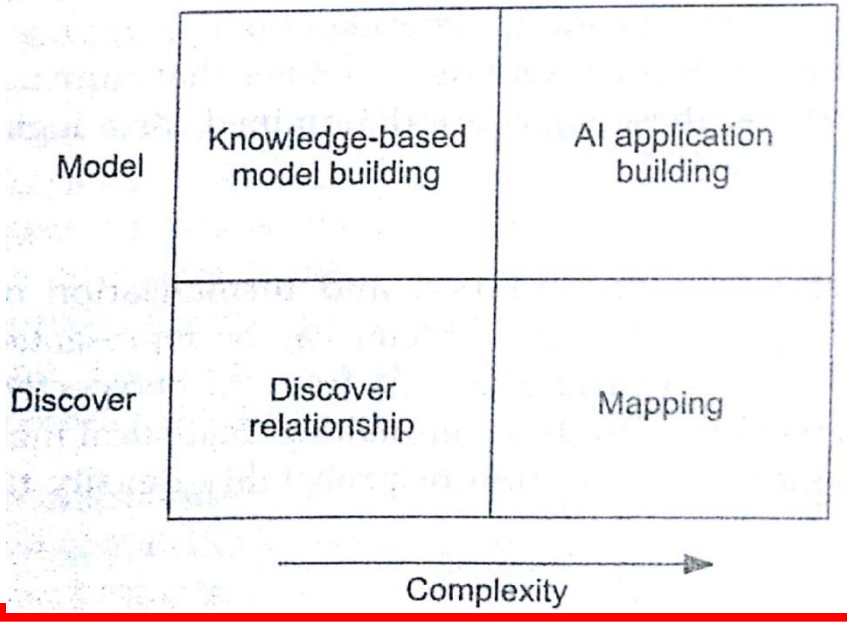
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

AI Models

- One important aspect of building AI solutions is modelling the problem.

(i) Dunker introduced 'Maze Hypothesis'.

(ii) logic theory machines.



AI Models

iii) Advent of NLP & man-machine dialogue.

iv) Formal models proposed to solve AI problems.

V) Human behaviour & psychological study-based inductive dynamic models for creative problem solving slowly became popular.

AI Models

- **Semiotic Models**

  - Based on Sign processes / signification and communication.

  - Code is specific which gives meaning to each sign based on the sound or letters that human use to form words or movements.

- **Statistical Models**

- Refers to representation and formulation of relationships through statistical techniques.

- Statistical model employs probabilistic approaches and is typically a collection of probability density function and distribution functions.

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI
- AI Models, **Data acquisition and learning aspects in AI**
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

Data acquisition and learning aspects in AI

Various AI –related topics on data acquisition and machine learning

- Knowledge discovery – Data mining and machine learning
- Computational learning theory (COLT)
- Neural and evolutionary computation
- Intelligent agents and multi-agent systems
- Multi-perspective integrated intelligence

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- **Problem solving- Problem solving process, Formulating problems**
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

Problem – Solving

Problem Solving – Is an area to deal with finding answer for some unknown situations.

It involves

- **Understanding**
- **Representation**
- **Formulation**
- **Solving**

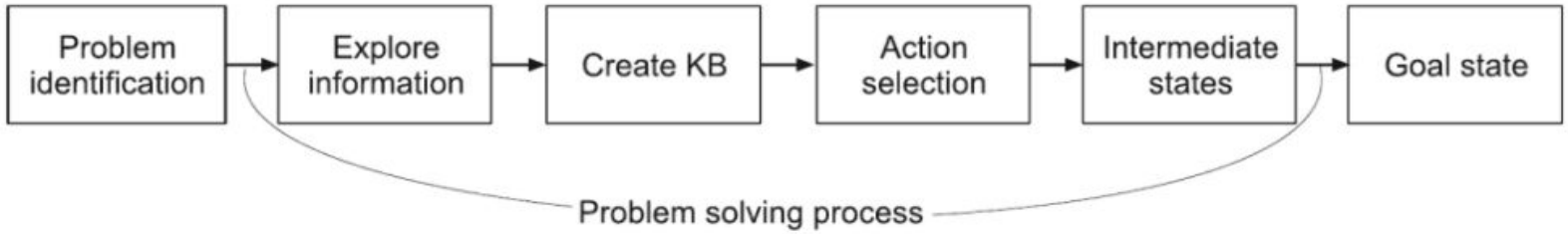
May be SIMPLE or COMPLEX Problems.

METHODS OF PROBLEM SOLVING:

1. General Purpose
2. Special purpose

Problem – Solving also assist in planning and decision making.

Problem – Solving Process



Example: ***Travelling in Romania***

Scenario : On holiday in Romania; currently in Arad Flight leaves tomorrow from Bucharest.

Goal : Be in Bucharest

Formulate problem :

States: various cities

Actions: drive between cities

Solution:

Appropriate sequence of cities

e.g.: Arad, Sibiu, Fagaras, Bucharest

Problem Solving with AI
" Formulate , Search , Execute " design for agent

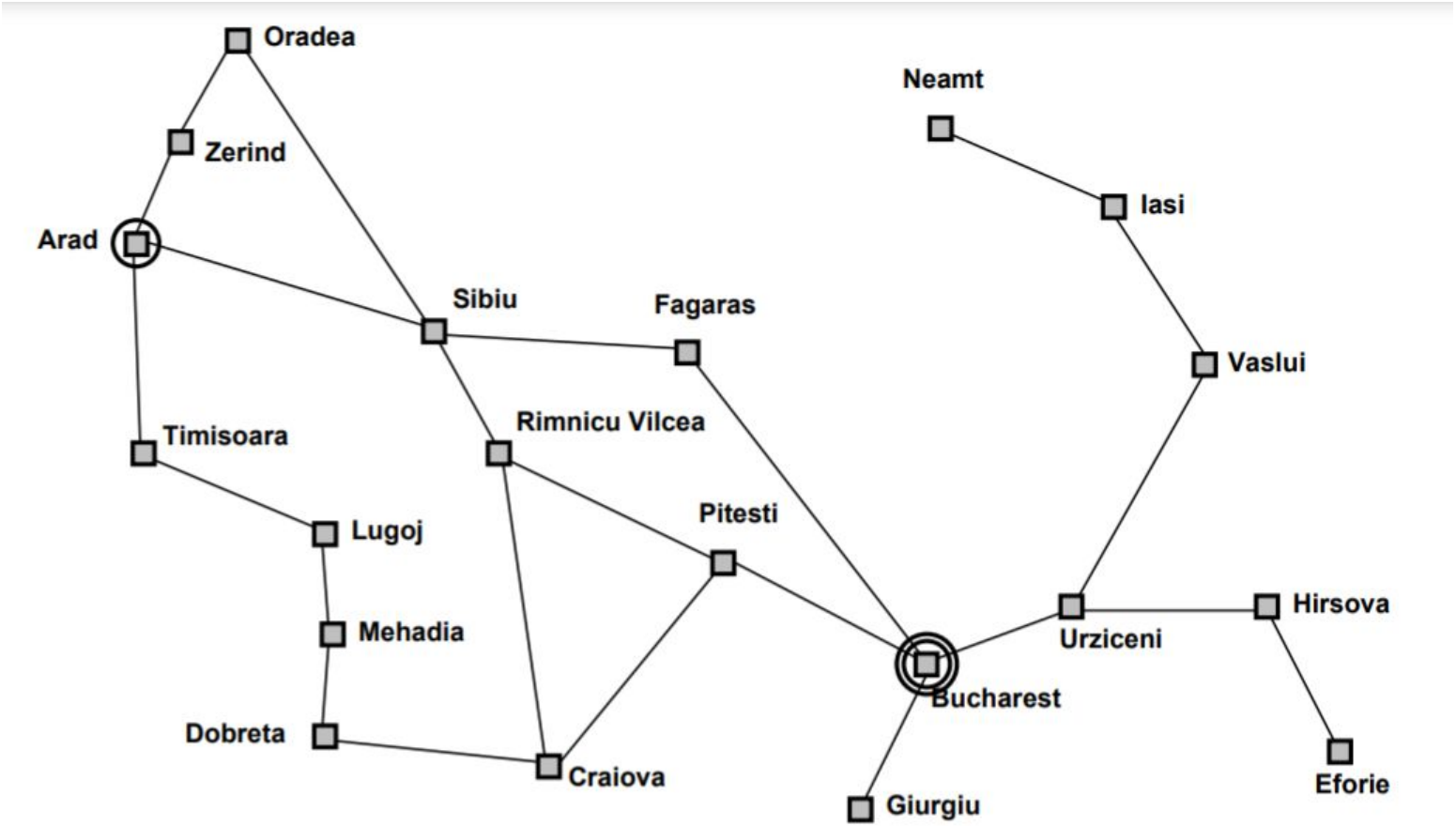

```
function SIMPLE-PROBLEM-SOLVING-AGENT(percept) returns an action
  persistent: seq, an action sequence, initially empty
              state, some description of the current world state
              goal, a goal, initially null
              problem, a problem formulation

  state ← UPDATE-STATE(state, percept)
  if seq is empty then
      goal ← FORMULATE-GOAL(state)
      problem ← FORMULATE-PROBLEM(state, goal)
      seq ← SEARCH(problem)
      if seq = failure then return a null action
  action ← FIRST(seq)
  seq ← REST(seq)
  return action
```

A simple problem-solving agent. It first formulates a goal and a problem, searches for a sequence of actions that would solve the problem, and then executes the actions one at a time. When this is complete, it formulates another goal and starts over.

Components of a problem – Example

A simplified road map of part of Romania.



Problem Solving with AI

Components of a problem

A **problem** can be defined formally by five components:

- The **initial state** that the agent starts in /Starting state which agent knows itself.
- Ex- The initial state for our agent in Romania might be described as In(Arad)

- A description of the possible **actions/operators** available to the agent. Given a particular state s, ACTIONS(s) returns the set of actions that can be executed in s. We say that each of these actions is **applicable** in s.
- Ex- from the state In(Arad), the applicable actions are {Go(Sibiu), Go(Timisoara), Go(Zerind)}.
- A description of what each action does; the formal name for this is the **transition model**, specified by a function RESULT(s, a) that returns the state that results from doing action a in state s. We also use the term **successor** to refer to any state reachable from a given state by a single action.
- Ex- RESULT(In(Arad),Go(Zerind)) = In(Zerind)

Components of a problem

- Together, the initial state, actions, and transition model implicitly define the **state space** of the problem—the set of all states reachable from the initial state by any sequence of actions. The state space forms a directed network or **graph** in which the nodes are states and the links between nodes are actions. A **path** in the state space is a sequence of states connected by a sequence of actions.
- Ex- The map of Romania shown can be interpreted as a state-space graph if we view each road as standing for two driving actions, one in each direction.
- The **goal test**, which determines whether a given state is a goal state. Sometimes there is an explicit set of possible goal states, and the test simply checks whether the given state is one of them.
- Ex- The agent's goal in Romania is the singleton set {In(Bucharest)}

Components of a problem

- A **path cost** function that assigns a numeric cost to each path. The problem-solving agent chooses a cost function that reflects its own performance measure.

- The **step cost** of taking action a to go from one state 's' to reach state 'y' is denoted by c(s, a, y).

Ex- For the agent trying to get to Bucharest, time is of the essence, so the cost of a path might be its length in kilometres. We assume that the **cost of a path** can be described as the *sum* of the costs of the individual actions along the path. The **step costs** for Romania are shown in Figure as route distances. We assume that step costs are nonnegative.

- A **solution** to a problem is an action sequence that leads from the initial state to a goal state. Solution quality is measured by the path cost function, and an **optimal solution** has the lowest path cost among all solutions.

Formulating Problems

- **Problem Formulation :** Choosing relevant set of states & feasible set of operators for moving from one state to another.

- **Search :** Is a process of imagining sequences of operators(actions) applied to initial state and to see which state reaches goal state.

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

Problem Types

1. Deterministic or observable (single-state)
2. Non-observable (multiple-state)
3. Non-deterministic or partially observable
4. Unknown state space

Problem Types

1. Deterministic or observable(Single-state problems)

- Each state is fully observable and it goes to one definite state after any action.
- Here , the goal state is reachable in one single action or sequence of actions.
- Deterministic environments ignore uncertainty.
- *Ex- Vacuum cleaner with sensor.*

Problem Types

2. Non-observable(Multiple-state problems) / conformant problems

- **Problem – solving agent does not have any information about the state.**
- **Solution may or may not be reached.**
- *Ex- In case of vacuum cleaner , the goal state is to clean the floor rather clean floor. Action is to suck if there is dirt. So , in non-observable condition , as there is no sensor , it will have to suck the dirt , irrespective of whether it is towards right or left . Here , the solution space is the states specifying its movement across the floor.*

Single-state

Start in: 5

Solution: [*right*, *suck*]

Multiple-state

Start in: $\{1,2,3,4,5,6,7,8\}$

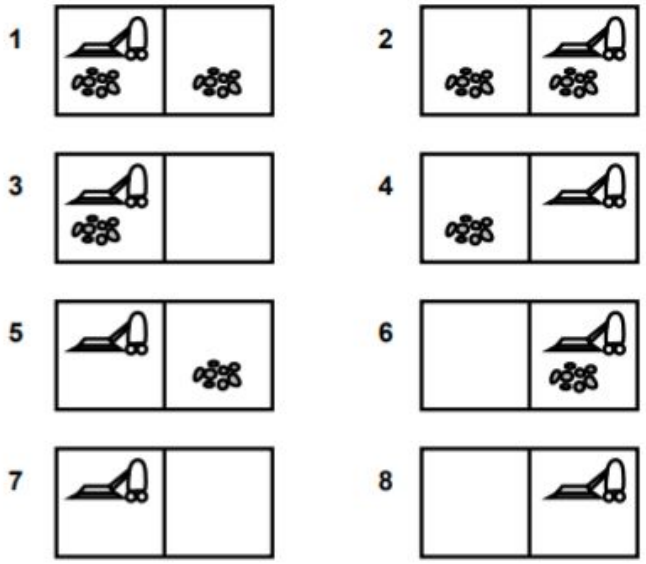
Solution: [*right*, *suck*, *left*, *suck*]

right $\rightarrow \{2,4,6,8\}$

suck $\rightarrow \{4,8\}$

left $\rightarrow \{3,7\}$

suck $\rightarrow \{7\}$

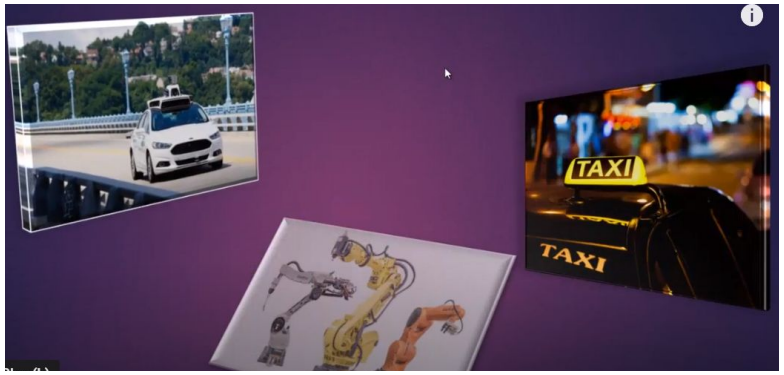


Problem Types

3. Non-deterministic(partially observable) problem

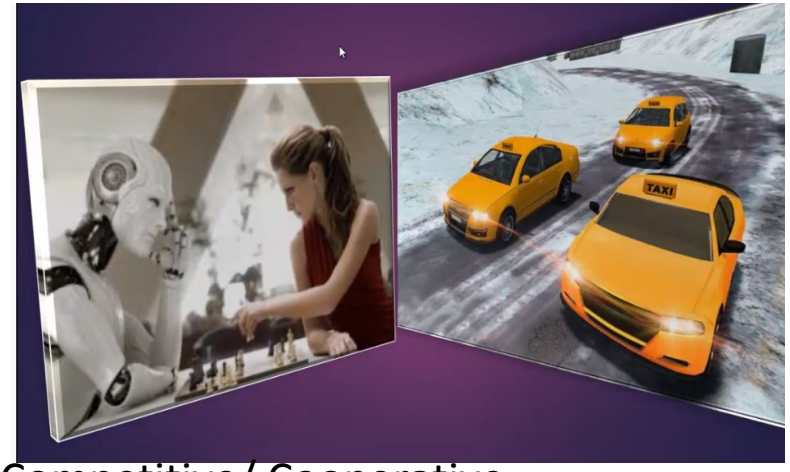
- **The effect of action is not clear.**
- **Percepts provide new information about the current state.**
- *Ex- If we take Vacuum cleaner , and now assume that the sensor is attached to it , then it will suck if there is dirt. Movement of the cleaner will be based on its current percept.*

Problem Types



Fully observable/ partially observable / unobservable

Single agent/ Multiagent



Competitive/ Cooperative

Static, Discrete: Crossword puzzle.
Dynamic, Continuous: Taxi Driving

Problem Types

4. Unknown state space problems

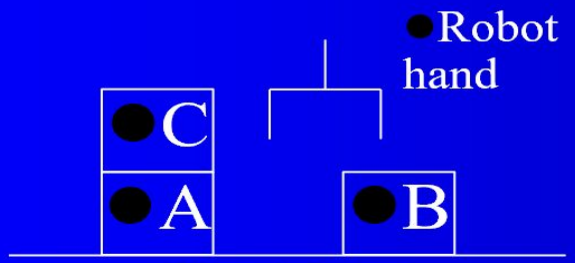
- **Typically exploration problems**
- **States and impact of actions are not known**
- ***Ex- online search that involves acting without compete knowledge of the next state or scheduling without map.***

Problem Characteristics

1. Is the problem decomposable ?
2. Can Solution steps be ignored or undone ?
3. Is the Universe Predictable?
4. Is a good solution absolute or relative ?
5. Is the solution a state or a path?
6. What is the role of knowledge?
7. Does the task require interaction with a person ?

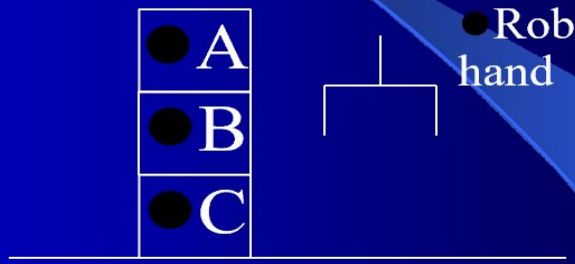
Problem Characteristics- 1. Is the problem decomposable ?
BLOCKS WORLD

- STRIPS : A planning system – Has rules with precondition deletion list and addition list



START

on(B, table)
on(A, table)
on(C, A)
hand empty
clear(C)
clear(B)

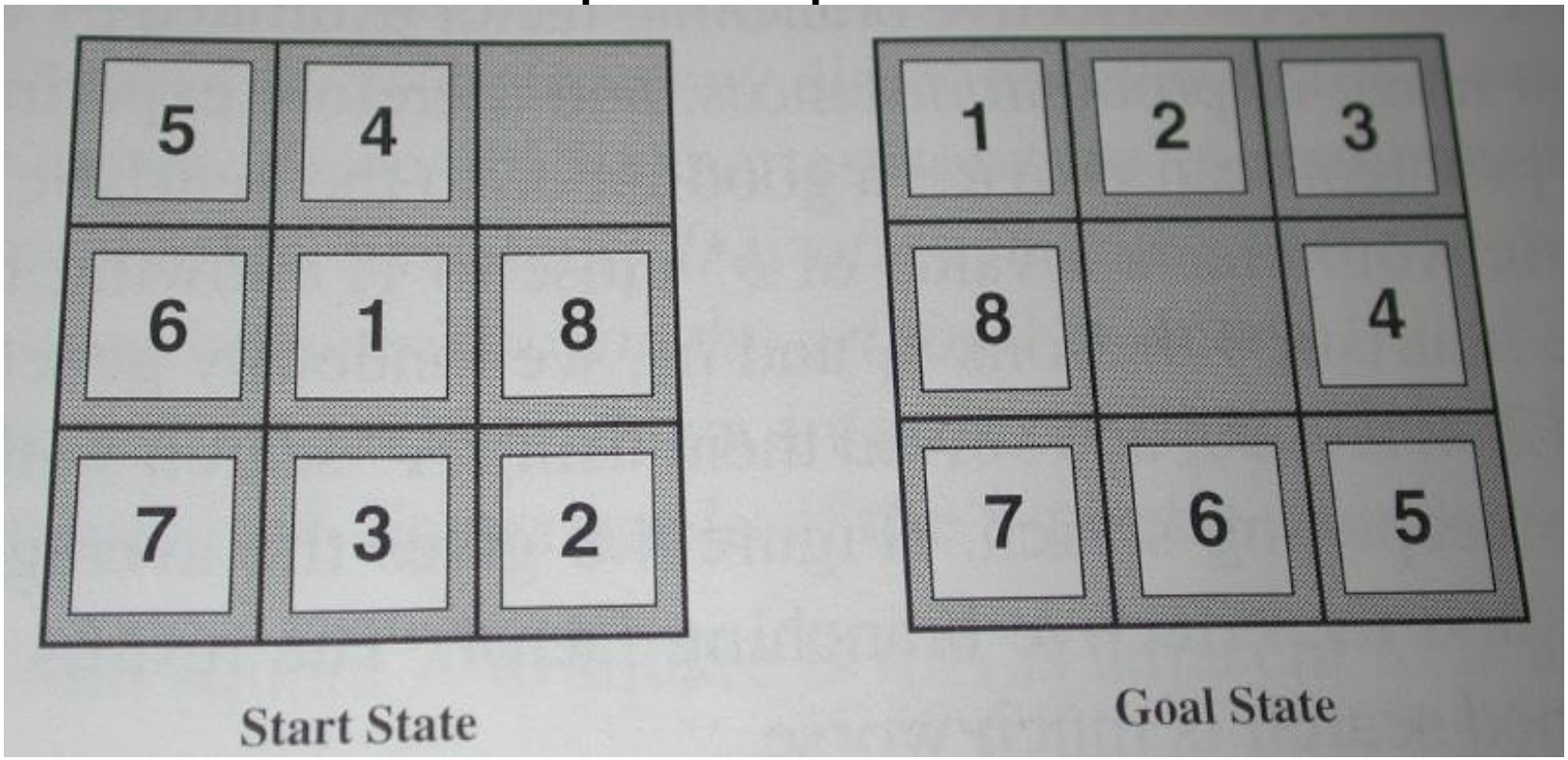


GOAL

on(C, table)
on(B, C)
on(A, B)
hand empty
clear(A)

Problem Characteristics: 2. Can Solution steps be ignored or undone ?
The 8 – Puzzle

- How would you use AI techniques to solve the 8-puzzle problem?



Important Classes of Problem

- Ignorable Ex. Theorem Proving
- Recoverable Ex. The 8-Puzzle
- Irrecoverable Ex. Chess

Problem Characteristics
3. Is the Universe Predictable?

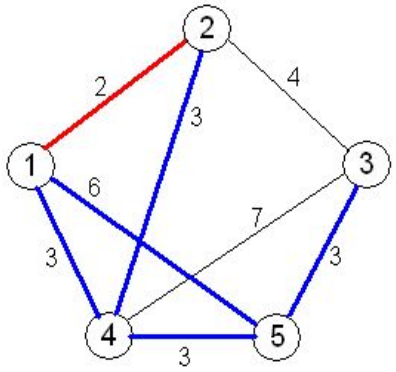
- Difference between **certain outcome** (e.g. 8-Puzzle) and **uncertain outcome** (e.g. play bridge) .
- In case of **certain outcome,** we can make a plan to generate a sequence of operation that guaranteed to lead to a solution. So, that the outcome is very certain.
- In case of **uncertain outcome** 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.

Problem Characteristics
4. Is a good solution absolute or relative ?

- The facts can be represented using a formal language called “Predicate Logic”.
- There may be “n” different solutions. If one solution is found , there is no need to go back and see if some path might also lead to a solution.
- Ex: P -> Q
- If marks = 92 -> Grade = O

RELATIVE SOLUTION : Ex. TSP

(Explore all possible solutions)



Problem Characteristics
5. Is the solution a state or a path ?

- To solve a problem, of finding interpretation, we need to produce only the interpretation itself. Record of processing by how the interpretation was arrived is NOT required.

Ex: The President ate Food with a fork.

- In contrast, if we must produce, the final state along with the path that we found to that state along with sequence of operations to produce the final state.

Problem Characteristics
6. What is the role of knowledge ?

Is the lots of knowledge required to solve a problem or is knowledge only to constrain solutions?

- Ex: Newspaper reading (by a computer) illustrate difference between problems for which a lot of knowledge is important only to constrain a search for solution and those for which a lot of knowledge is required to recognise a solution.
- Ex: Scanning newspapers to find who support Party A or Party B in upcoming elections.
- Assume unlimited computer power.
- The following has to be known at the time to arrive at an answer
  - (i) Name of Party candidates
  - (ii) Taxes lowered
  - (iii) Improved education, etc.

Problem Characteristics

7. Does the task require interaction with a person ?

There are 2 types of problems:

- Solitary : in which computer is given a problem description and with NO DEMAND FOR EXPLANATION of the reasoning process.
- Conversational: in which there is intermediate communication between a person and the computer either to provide additional assistance to computer or to provide additional information to user or both.

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

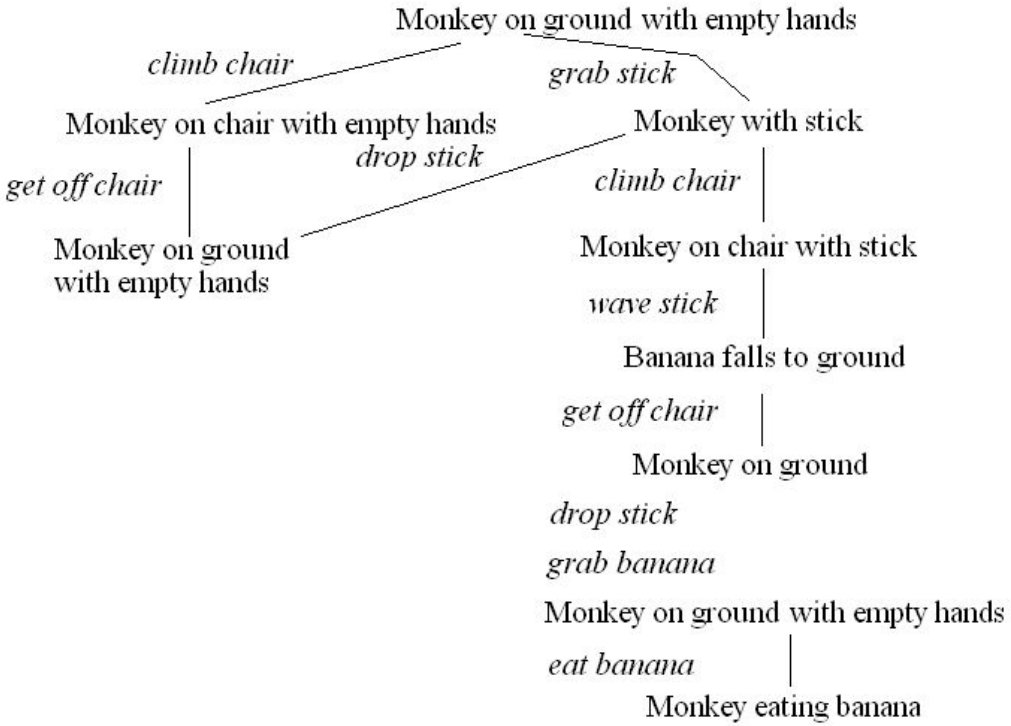
- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

Formal Description of a Problem

- In AI, we will formally define a problem as
  - ***a space of all possible configurations where each configuration is called a state***
    - ***thus, we use the term state space***
  - ***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 (see monkey & banana problem on the next slide)
  - 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

The Monkey & Bananas Problem

- A monkey is in a cage and bananas are suspended from the ceiling, the monkey wants to eat a banana but cannot reach them
  - in the room are a chair and a stick
  - if the monkey stands on the chair and waves the stick, he can knock a banana down to eat it
  - what are the actions the monkey should take?



Initial state:
monkey on ground
with empty hand
bananas suspended
Goal state:
monkey eating
Actions:
climb chair/get off
grab X

- • Problem solving: The term, Problem Solving relates to analysis in AI. Problem solving may be characterized as a systematic search through a range of possible actions to reach some predefined goal or solution. Problem-solving methods are categorized as special purpose and general purpose.
- • A special-purpose method is tailor-made for a particular problem, often exploits very specific features of the situation in which the problem is embedded.
- • A general-purpose method is applicable to a wide variety of problems. One General-purpose technique used in AI is ‘means-end analysis’: a step-bystep, or incremental, reduction of the difference between current state and final goal.

Example – Toy Problems
Vacuum Cleaner World

Program implements the agent function tabulated

Function Reflex-Vacuum-Agent([*location,status*]) return an action

If *status* = *Dirty* **then return** *Suck*

else if *location* = *A* **then return** *Right*

else if *location* = *B* **then return** *left*

Toy Problems vs Real-world Problems

- A **toy problem** is intended to illustrate or exercise various problem solving methods. It can be given a concise, exact description.

- A **real world problem** is one whose solutions people actually care about. Such problems tend not to have a single agreed-upon description, but we can give the general flavor of their formulations.

Toy Problem- 1

Vacuum Cleaner World -Problem Formulation

- State - agent is in one of two locations, each of which might or might not contain dirt. Thus there are $2 \times 2^2 = 8$ possible world states. A larger environment with n locations has $n \cdot 2^n$ states

- Initial State

– Any one of 8 states

- Actions

– In this simple environment, each state has just three actions: *Left* , *Right* ,*Suck*. Larger environments might also include *Up* , *Down*

Single-state

Start in: 5

Solution: [*right*, *suck*]

Multiple-state

Start in: $\{1,2,3,4,5,6,7,8\}$

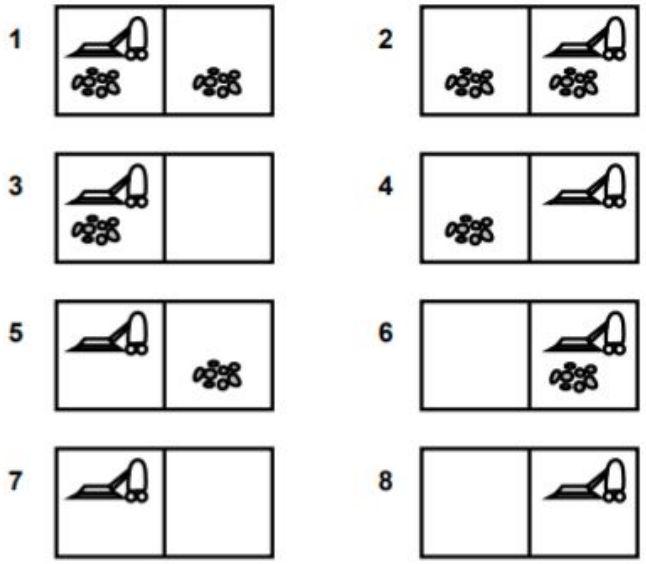
Solution: [*right*, *suck*, *left*, *suck*]

right $\rightarrow \{2,4,6,8\}$

suck $\rightarrow \{4,8\}$

left $\rightarrow \{3,7\}$

suck $\rightarrow \{7\}$



Toy Problem- 1

Vacuum Cleaner World -Problem Formulation

- **Transition model**: The actions have their expected effects, except that moving *Left* in the leftmost square, moving *Right* in the rightmost square, and *Suck*ing in a clean square have no effect. The complete state space is shown in the figure .

- State Space for the Vacuum World

Labels on Arcs denote L: Left, R: Right, S: Suck

- Goal Test
  - This checks whether all the squares are clean
- Path Cost
  - Number of steps (each step costs a value of 1)

Toy Problem- 2
The 8-Puzzle (Sliding Block Puzzle)

The **8-puzzle**, an instance of which is shown below, consists of a 3×3 board with eight numbered tiles and a blank space. A tile adjacent to the blank space can slide into the space. The object is to reach a specified goal state, such as the one shown on the right of the figure.

1	2	3
7	8	4
6		5

Initial State

1	2	3
8		4
7	6	5

Goal State

Toy Problem- 2
The 8-Puzzle (Sliding Block Puzzle)

States: A state description specifies the location of each of the eight tiles and the blank in one of the nine squares.

Initial state: Any state can be designated as the initial state.

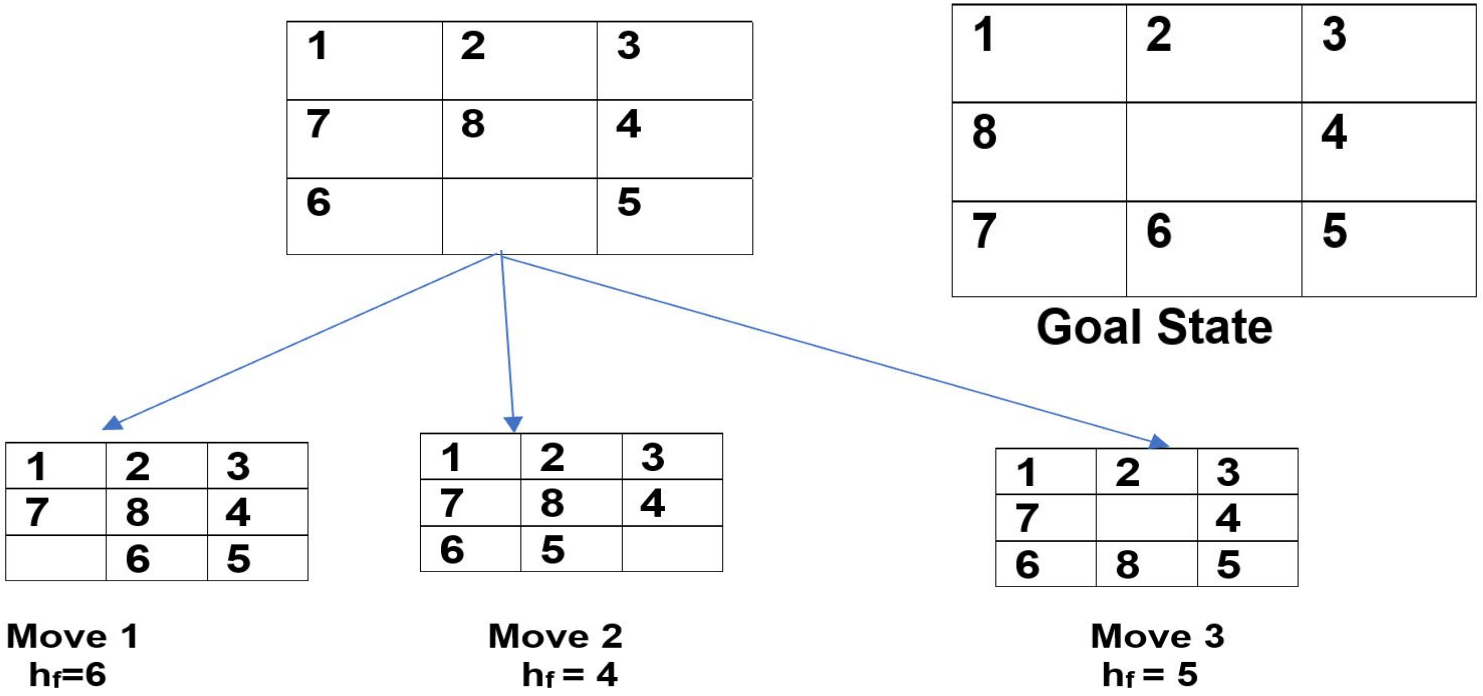
Successor function: This generates the legal states that result from trying the four actions (blank moves Left, Right, Up, or Down).

Goal test: This checks whether the state matches the goal configuration (Other goal configurations are possible.)

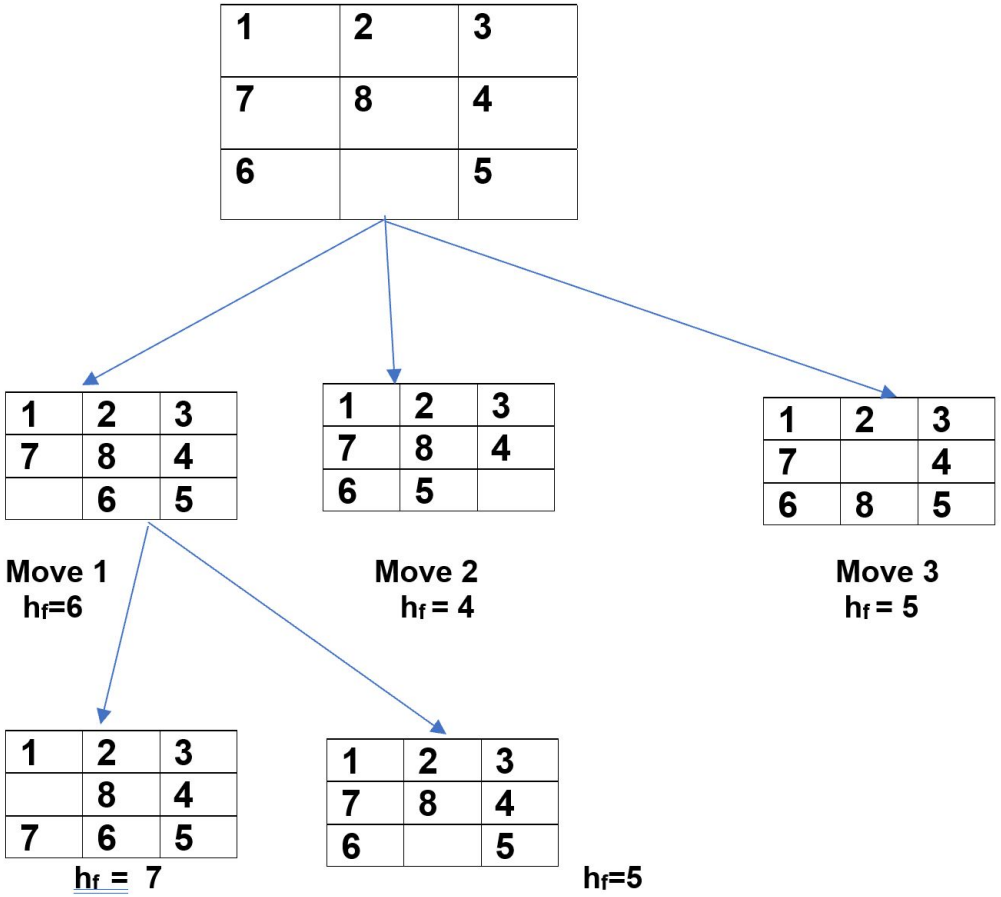
Path cost: Each step costs 1, so the path cost is the number of steps in the path.

Toy Problem- 2
The 8-Puzzle (Sliding Block Puzzle) - Solution

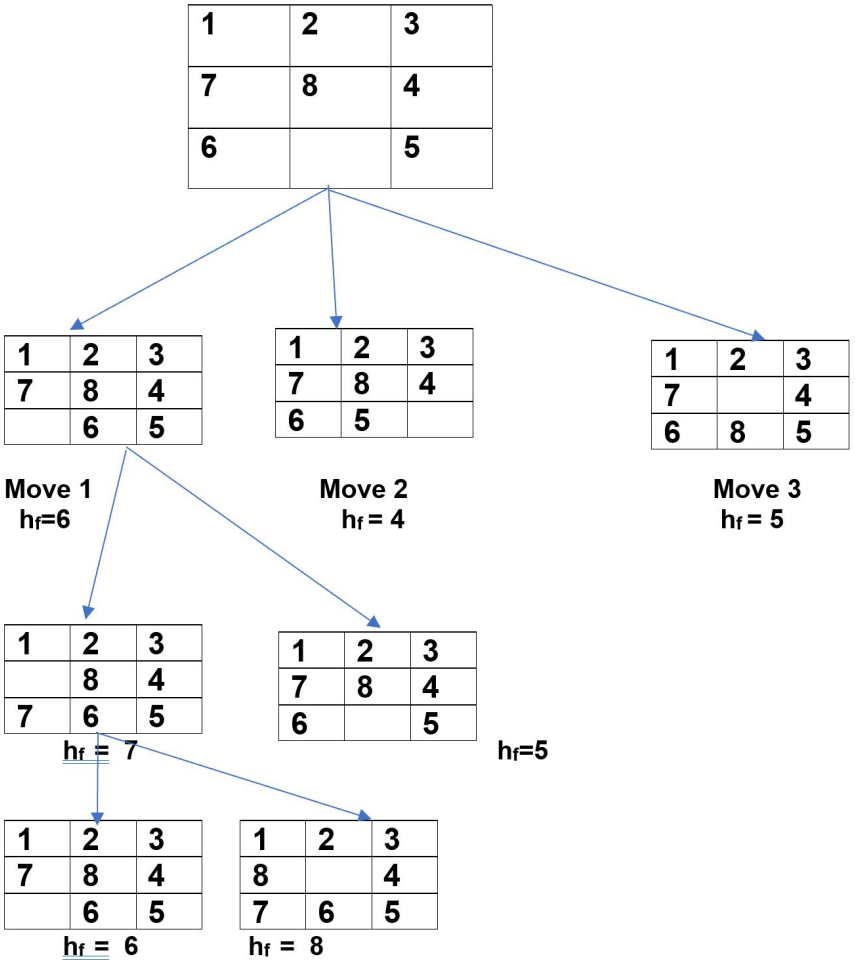
- h_f= **+1** for every correct position
- Solution of this problem is “movement of tiles” in order to reach goal state.
- The transition function or legal move is any one tile movement by one space in any direction.



Toy Problem- 2
The 8-Puzzle (Sliding Block Puzzle) - Solution



Toy Problem- 2
The 8-Puzzle (Sliding Block Puzzle) - Solution



Toy Problem- 3

Water – Jug Problem

A Water Jug Problem: You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug?

Toy Problem- 3

Water – Jug Problem

Solution:

The state space for this problem can be described as the set of ordered pairs of integers **(x,y)**

Where,

X represents the quantity of water in the 4-gallon jug **X= 0,1,2,3,4**
Y represents the quantity of water in 3-gallon jug **Y=0,1,2,3**

Note

$0 \leq X \leq 4$, and $0 \leq Y \leq 3$

Start State: (0,0)

Goal State: **(2, n)** for any n. Attempting to end up in a goal state.(since the problem doesn't specify the quantity of water in 3-gallon jug)

Toy Problem- 3

Water – Jug Problem

Generate production rules for the water jug problem

Production Rules:

1.	(x,y) -> (4,y)	Fill x
2.	(x,y) -> (x,3)	Fill y
3.	(x,y) -> (x-d, y)	Pour water out from X
4.	(x,y) -> (x,y-d)	Pour water from y
5.	(x,y) -> (0,y)	Empty x
6.	(x,y) -> (x,0)	Empty y
7.	(x,y) -> (4,y-(4-x))	Pour water from y into x until x is full
8.	(x,y) -> (x – (3-y), 3)	Pour water from x into y until y is full.
9.	(x,y) -> (x+y, 0)	Pour all water from y to x
10.	(x,y) -> (0, x+y)	Pour all water from x to y
11.	(0,2) -> (2,0)	Pour 2 Gallon of water from y to x
12.	(2, y) -> (0,y)	Pour 2 Gallon of water from x to ground.

Toy Problem- 3

Water – Jug Problem

First solution

	4-g jug	3-g jug
Initial	0	0
R2	0	3
R9	3	0
R2	3	3
R7	4	2
R5	0	2
R9	2	0

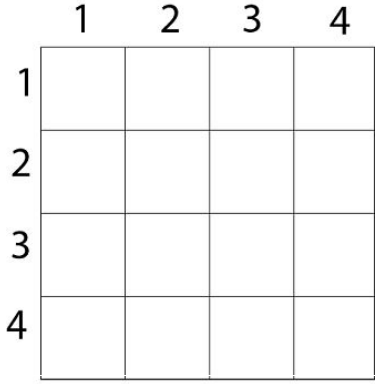
Goal State

1. (x,y) -> (4,y) Fill x
2. (x,y) -> (x,3) Fill y
3. (x,y) -> (x-d, y) Pour water out from X
4. (x,y) -> (x,y-d) Pour water from y
5. (x,y) -> (0,y) Empty x
6. (x,y) -> (x,0) Empty y
7. (x,y) -> (4,y-(4-x)) Pour water from y into x until x is full
8. (x,y) -> (x – (3-y), 3) Pour water from x into y until y is full.
9. (x,y) -> (x+y, 0) Pour all water from y to x
10. (x,y) -> (0, x+y) Pour all water from x to y
11. (0,2) -> (2,0) Pour 2 Gallon of water from y to x
12. (2, y) -> (0,y) Pour 2 Gallon of water from x to ground.

Toy Problem- 4(a)
4-queens problem

The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other.

Given a 4 x 4 chessboard and number the rows and column of the chessboard 1 through 4.

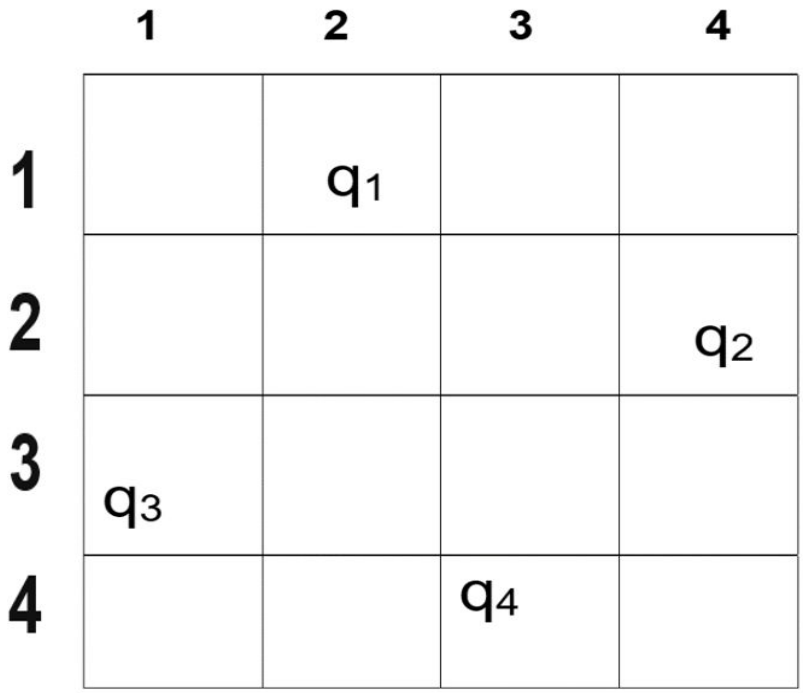


4x4 chessboard

Since, we have to place 4 queens such as q_1 q_2 q_3 and q_4 on the chessboard, such that no two queens attack each other. In such a conditional each queen must be placed on a different row, i.e., we put queen "i" on row "i."

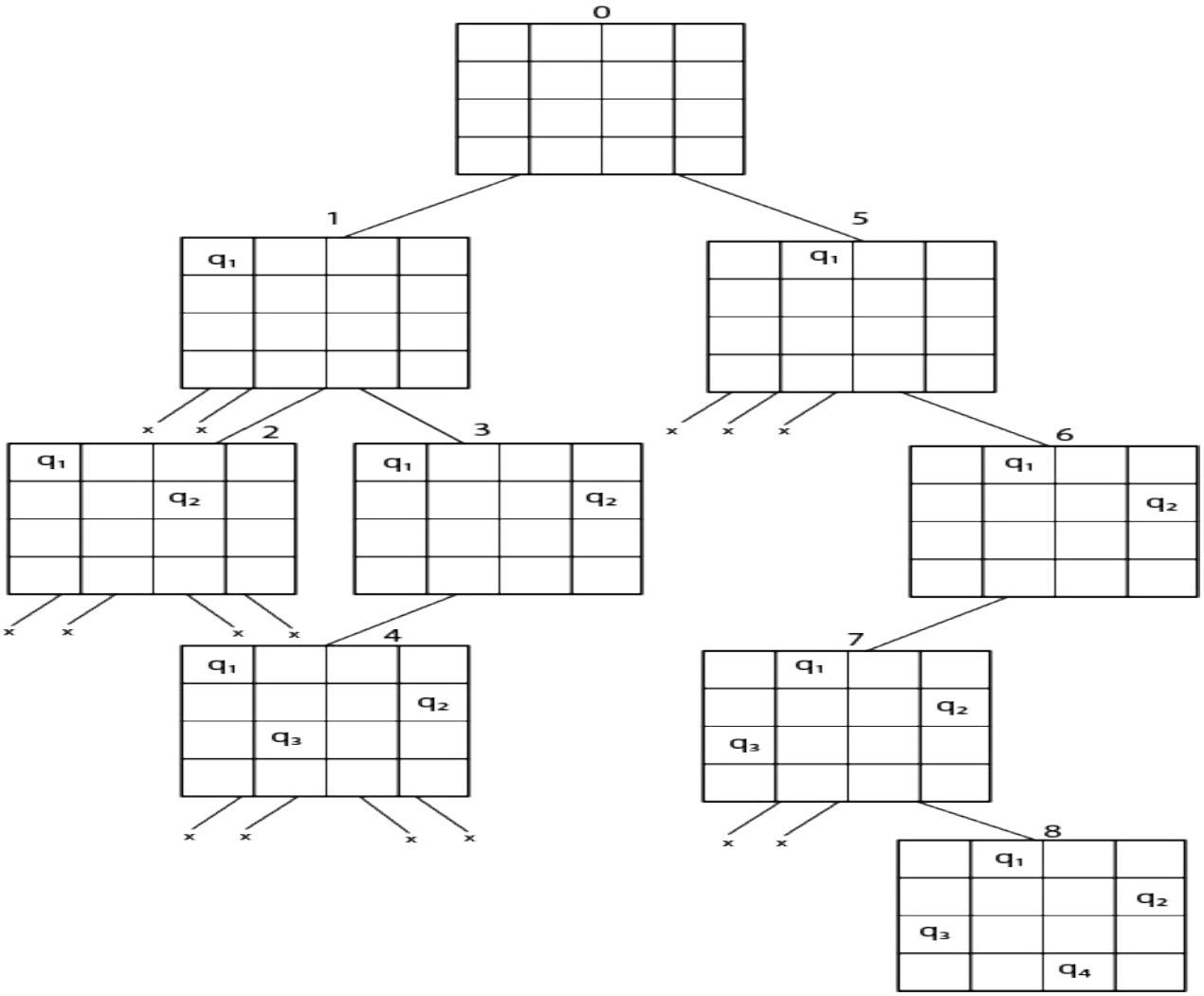
Toy Problem- 4(a)
4-queens problem

One possible solution for the 4-queens problem is (2,4,1,3) i.e ,



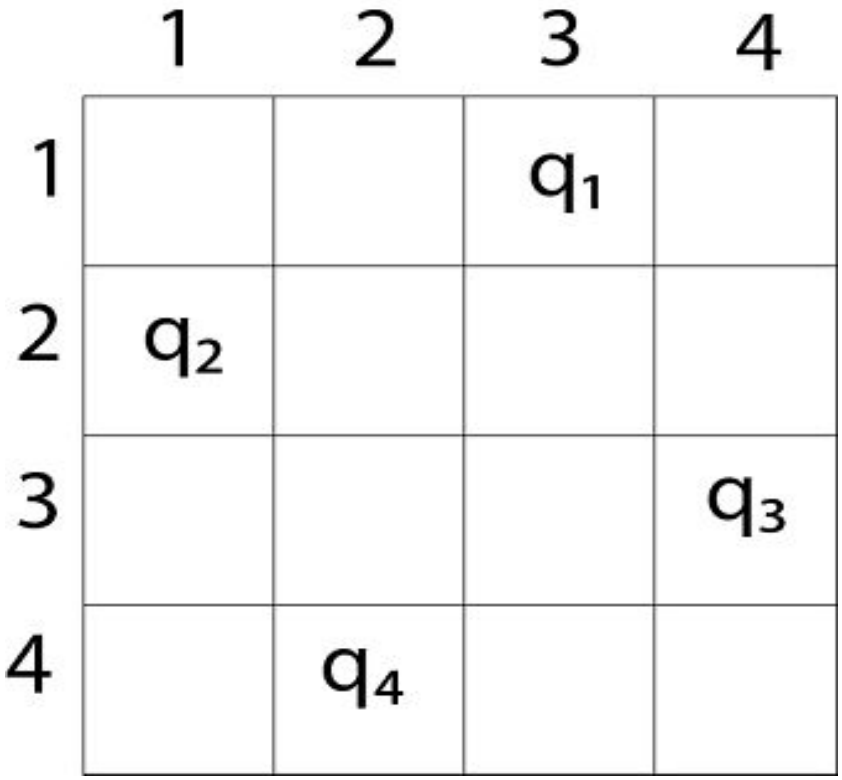
Toy Problem- 4(a)
4-queens problem

The implicit tree for 4 - queen problem for a solution (2, 4, 1, 3) is as follows:



Toy Problem- 4(a)
4-queens problem

Another solution for 4 - queens problems is (3, 1, 4, 2) i.e



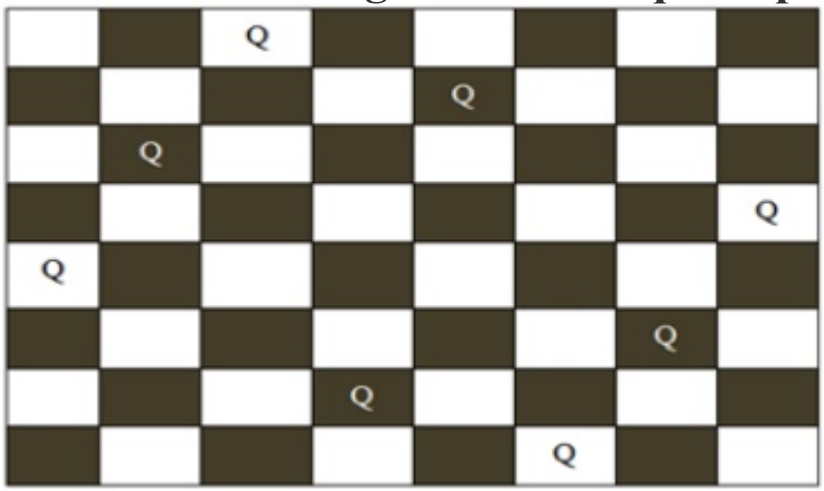
Toy Problem- 4(b)
8-queens problem

"We have 8 queens and an 8x8 Chess board having alternate black and white squares. The queens are placed on the chessboard.

Any queen can attack any other queen placed on same row, or column or diagonal. We have to find the proper placement of queens on the Chess board in such a way that no queen attacks other queen".

Toy Problem- 4(b)
8-queens problem

possible board configuration of 8 queen problem



In figure , the possible board configuration for 8-queen problem has been shown. The board has alternative black and white positions on it. The different positions on the board hold the queens. The production rule for this game is you cannot put the same queens in a same row or same column or in same diagonal. After shifting a single queen from its position on the board, the user have to shift other queens according to the production rule. Starting from the first row on the board the queen of their corresponding row and column are to be moved from their original positions to another position. Finally the player has to be ensured that no rows or columns or diagonals of on the table is same.

Toy Problem- 4(b)
8-queens problem

The first incremental formulation one might try is the following:

- **States**: Any arrangement of 0 to 8 queens on the board is a state.
- **Initial state**: No queens on the board.
- **Actions/Successor function** : Add a queen to any empty square.
- **Transition model**: Returns the board with a queen added to the specified square.
- **Goal test**: 8 queens are on the board, none attacked.
- **Path cost:** Zero (search cost only exists)

In this formulation, we have $64 \cdot 63 \cdot \cdots 57 \approx 1.8 \times 10^{14}$ possible sequences to investigate.

Toy Problem- 5
BLOCK WORLD

What is the Blocks World? -- The world consists of:

- A flat surface such as a tabletop
- An adequate set of identical blocks which are identified by letters.
- The blocks can be stacked one on one to form towers of apparently unlimited height.
- The stacking is achieved using a robot arm which has fundamental operations and states which can be assessed using logic and combined using logical operations.
- The robot can hold one block at a time and only one block can be moved at a time.

Toy Problem- 5

Blocks world

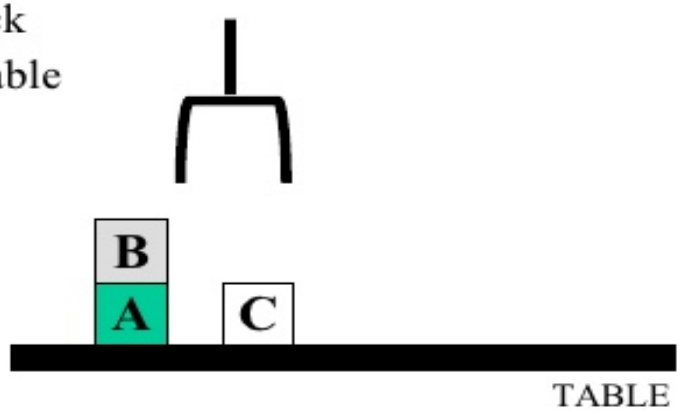
The **blocks world** is a micro-world that consists of a table, a set of blocks and a robot hand.

Some domain constraints:

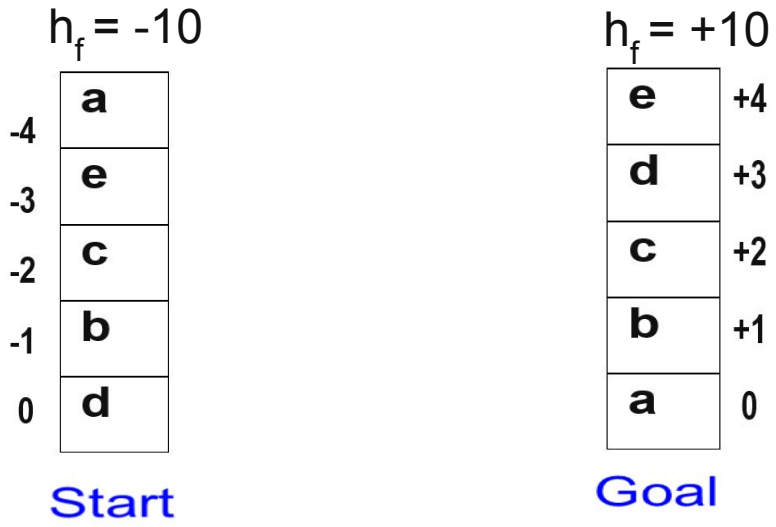
- Only one block can be on another block
- Any number of blocks can be on the table
- The hand can only hold one block

Typical representation:

ontable(a)
ontable(c)
on(b,a)
handempty
clear(b)
clear(c)



Toy Problem- 5
Blocks World Problem – Ex .



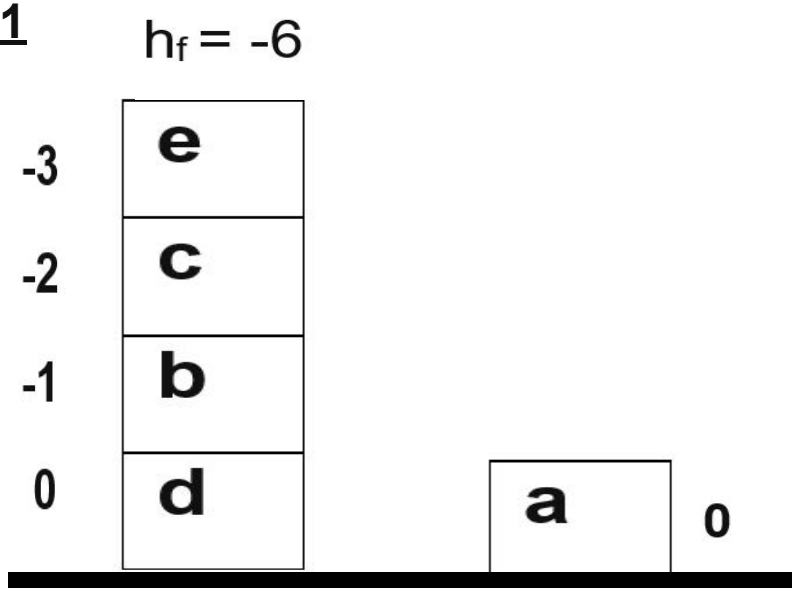
Heuristic

For each block that has the correct support structure: +1 to every block in the support structure.

For each block that has a wrong support structure: -1 to every block in the support structure.

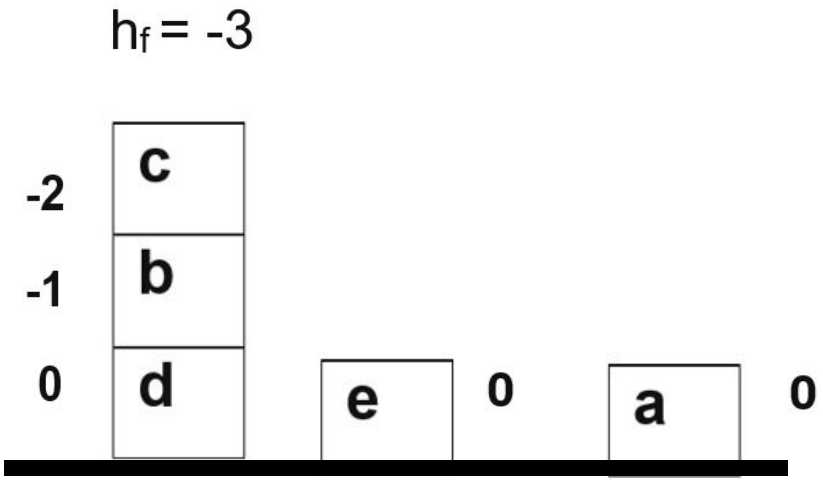
Toy Problem- 5
Blocks World Problem – Ex .

Step 1



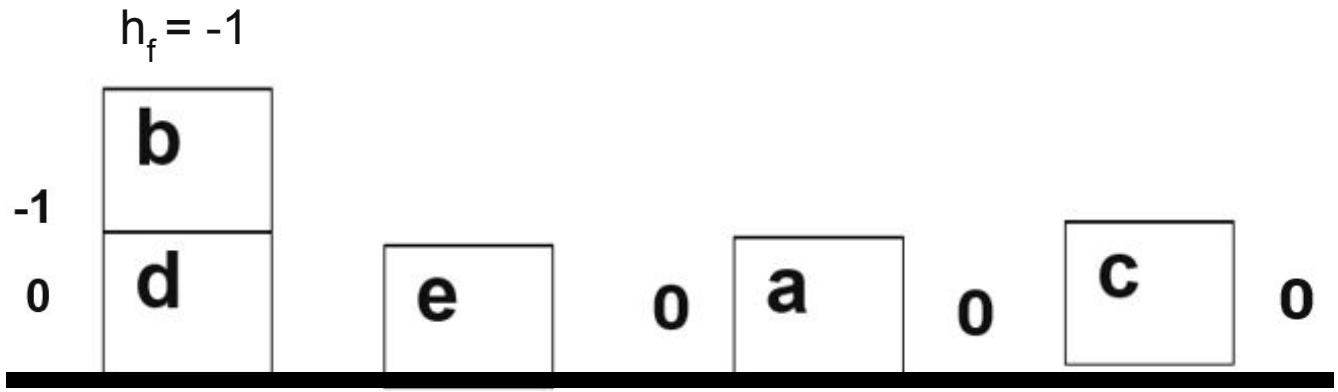
Toy Problem- 5
Blocks World Problem – Ex .

Step 2



Toy Problem- 5
Blocks World Problem – Ex .

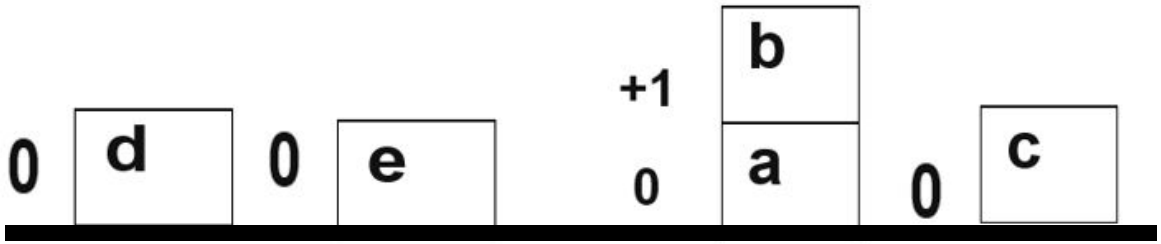
Step 3



Toy Problem- 5
Blocks World Problem – Ex .

Step 4

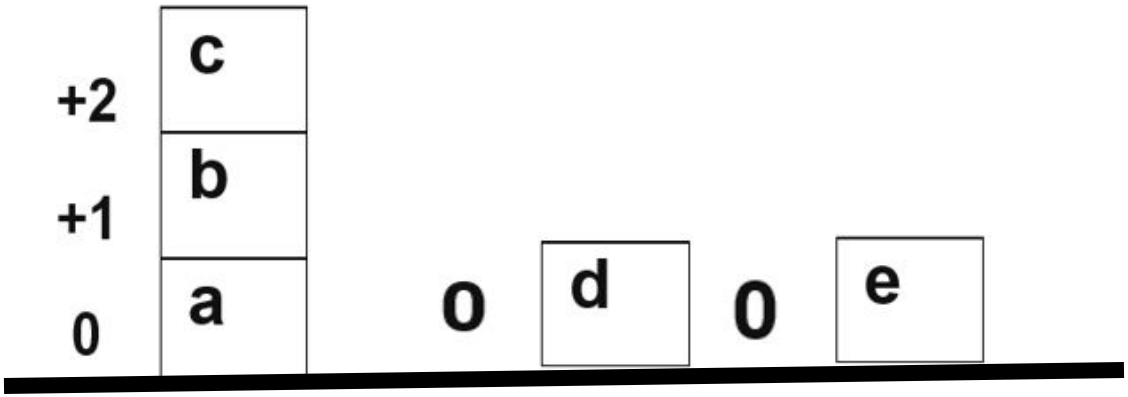
$h_f = +1$



Toy Problem- 5
Blocks World Problem – Ex .

Step 5

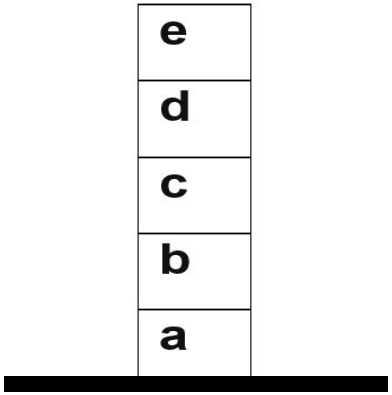
$h_f = +3$



Toy Problem- 5
Blocks World Problem – Ex .

Step 6

$h_f = +10$



Global solution for block world

Toy Problem- 5

BLOCK WORLD - STRIPS

(STanford Research Institute Problem Solver)

- STRIPS - an action-centric representation ,for each action , specifies the effect of an action.

A **STRIPS planning problem** specifies:

1) an initial state *S*
2) a goal *G*
3) a set of STRIPS actions

Toy Problem- 5

BLOCK WORLD - STRIPS

(STanford Research Institute Problem Solver)

- STRIPS - an action-centric representation ,for each action , specifies the effect of an action.

The STRIPS representation for an action consists of three lists,

- Pre_Cond list contains predicates which have to be true before operation.
- ADD list contains those predicates which will be true after operation
- DELETE list contain those predicates which are no longer true after operation

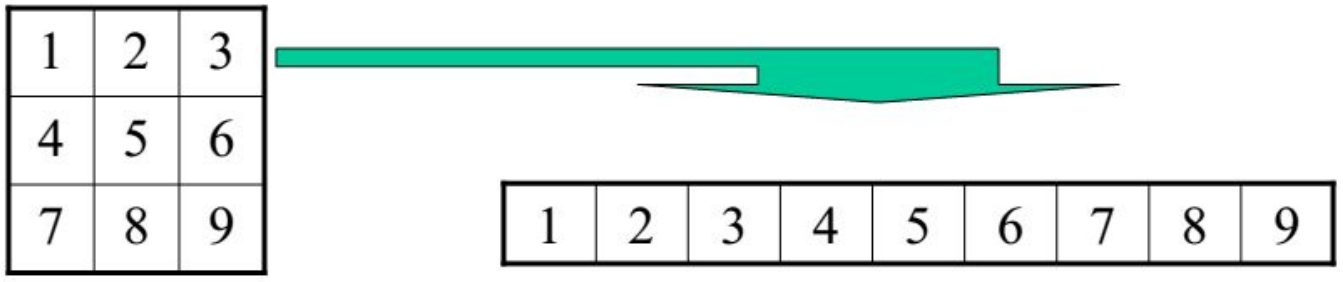
Toy Problem- 6
Tic Tac Toe

The game **Tic Tac Toe** is also known as **Noughts** and **Crosses** or **X**s and **O**s ,the player needs to take turns marking the spaces in a 3x3 grid with their own marks,
if 3 consecutive marks (**Horizontal**, **Vertical**,**Diagonal**) are formed then the player who owns these moves get won.

Assume ,

Player 1 - X
Plaver 2 - O

So,a player who gets 3 consecutive marks first,they will win the game .



2-D game-board

1-D Vector

Toy Problem- 7
Missionaries and Cannibals

Let Missionary is denoted by 'M' and Cannibal, by 'C'.
These rules are described below:

Rule 1 :	(0, M)	:	One missionary sailing the boat from bank-1 to bank-2
Rule 2 :	(M, 0)	:	One missionary sailing the boat from bank-2 to bank-1
Rule 3 :	(M, M)	:	Two missionaries sailing the boat from bank-1 to bank-2
Rule 4 :	(M, M)	:	Two missionaries sailing the boat from bank-2 to bank-1
Rule 5 :	(M, C)	:	One missionary and one Cannibal sailing the boat from bank-1 to bank-2
Rule 6 :	(C, M)	:	One missionary and one Cannibal sailing the boat from bank-2 to bank-1
Rule 7 :	(C, C)	:	Two Cannibals sailing the boat from bank-1 to bank-2
Rule 8 :	(C, C)	:	Two Cannibals sailing the boat from bank-2 to bank-1
Rule 9 :	(0, C)	:	One Cannibal sailing the boat from bank-1 to bank-2
Rule 10 :	(C, 0)	:	One Cannibal sailing the boat from bank-2 to bank-1

All or some of these production rules will have to be used in a particular sequence to find the solution of the problem.

Toy Problem- 7
Missionaries and Cannibals

Rules applied and their sequence in Missionaries and Cannibals problem

After application of rule	persons in the river bank-1	persons in the river bank-2	boat position
Start state	M, M, M, C, C, C	0	bank-1
5	M, M, C, C	M, C	bank-2
2	M, M, C, C, M	C	bank-1
7	M, M, M	C, C, C	bank-2
10	M, M, M, C	C, C	bank-1
3	M, C	C, C, M, M	bank-2
6	M, C, C, M	C, M	bank-1
3	C, C	C, M, M, M	bank-2
10	C, C, C	M, M, M	bank-1
7	C	M, M, M, C, C	bank-2
10	C, C	M, M, M, C	bank-1
7	0	M, M, M, C, C, C	bank-2

Toy Problem- 7
Formalization of the M&C Problem

State space: triple (x,y,z) with $0 \leq x,y,z \leq 3$, where x,y, and z represent the number of missionaries, cannibals and boats currently on the original bank.
Initial State: (3,3,1)
Successor function: From each state, either bring one missionary, one cannibal, two missionaries, two cannibals, or one of each type to the other bank.
Note: Not all states are attainable (e.g., (0,0,1)), and some are illegal.
Goal State: (0,0,0) Path Costs: 1 unit per crossing

Toy Problem- 8
Travelling Salesman Problem(Path Finding Problems)

"Given 'n' cities connected by roads, and distances between each pair of cities. A sales person is required to travel each of the cities exactly once. We are required to find the route of salesperson so that by covering minimum distance, he can travel all the cities and come back to the city from where the journey was started".

Diagrammatically, it is shown below

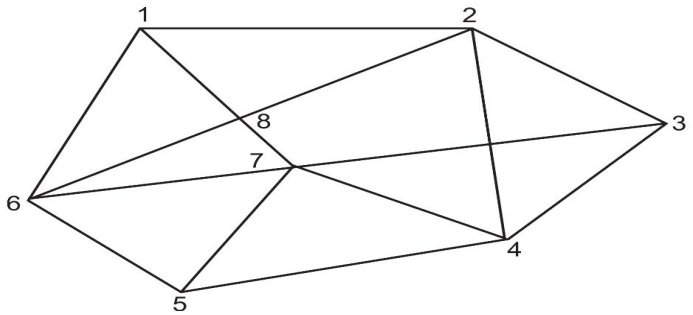


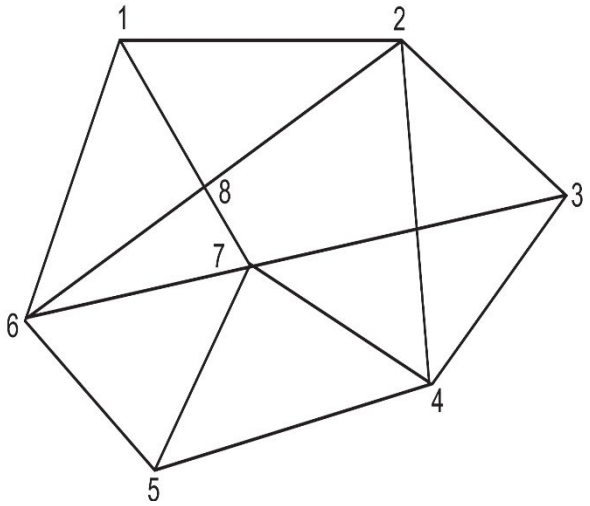
Fig : Cities and paths connecting these

Toy Problem- 8
Travelling Salesman Problem(Path Finding Problems)

The basic travelling salesperson problem comprises of computing the shortest route through a given set of cities.

Following Table shows number of cities and the possible routes mentioned against them.

Number of cities	Possible Routes
1	1
2	1 -2-1
3	1 -2 -3 1
	1 -3 -2 1
4	1- 2- 3- 4-1
	1- 2- 4- 3- 1
	1- 3- 2- 4- 1
	1- 3- 4- 2- 1
	1- 4- 2- 3-1
	1- 4- 3- 2- 1



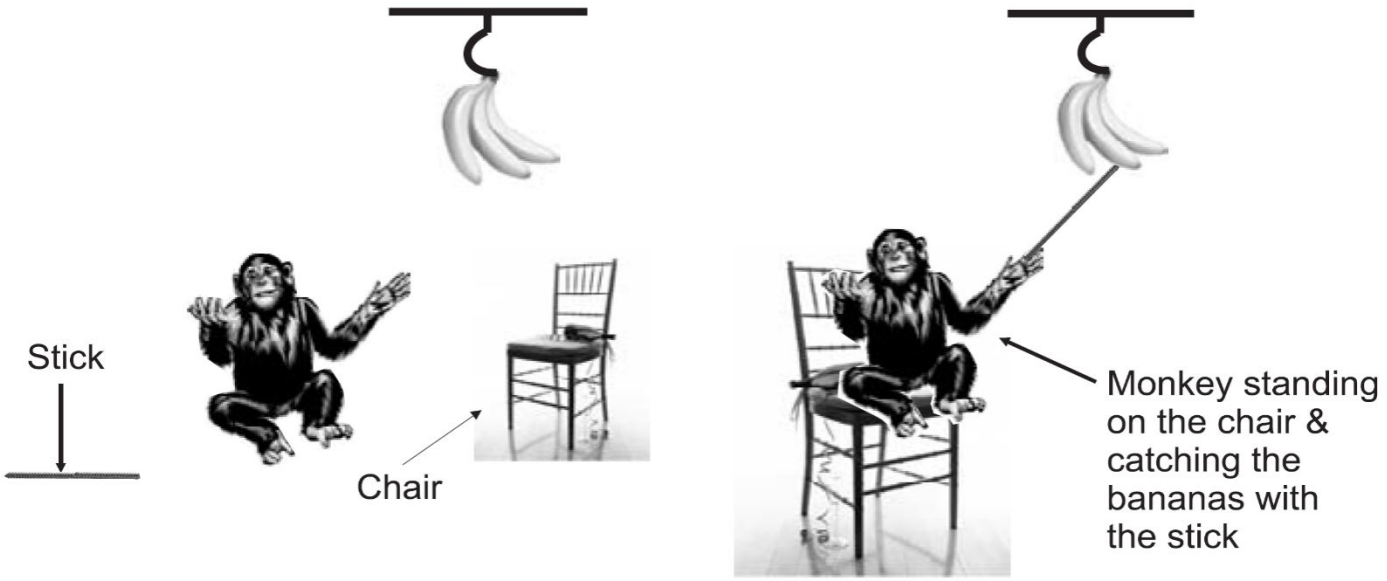
Toy Problem- 9
Monkey Banana Problem

"A monkey is in a room. A bunch of bananas is hanging from the ceiling. The monkey cannot reach the bananas directly. However , in the room there is one chair and a stick. The monkey can reach the banana standing on the chair. We have to find the sequence of events by which monkey can reach the bananas."

Toy Problem- 9
Monkey Banana Problem

Solution of this problem means finding the sequence of actions for the monkey to reach the banana.

Monkey standing on the chair and catching the bananas with the stick.



Summary of Problem Solving with AI – Toy Problems

1. Block World
2. 4 Queens/ 8 Queens
3. Tic Tac Toe
4. Water Jug
5. Monkey Banana
6. 8 Puzzle
7. TSP
8. Vacuum Cleaner
9. Missionaries and Cannibals

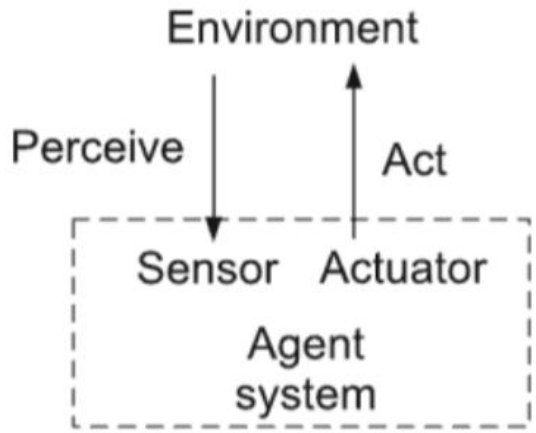
Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

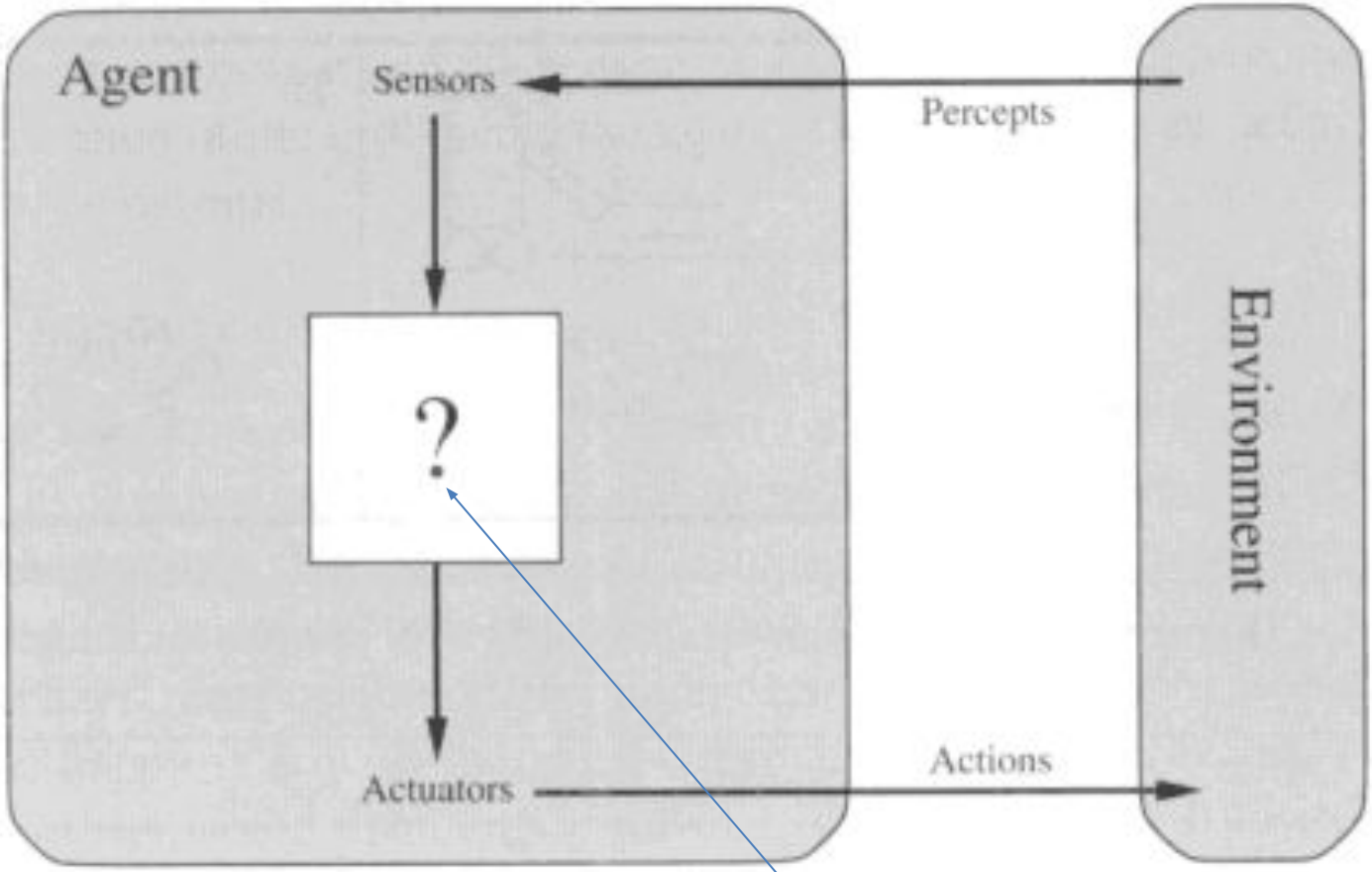
Agent

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



Basic structure of an agent.

Diagram of an Agent



What AI should fill

Intelligent Agents

Intelligent Agent – Is an entity that works without assistance, interprets the input, senses the environment, makes choices and ultimately acts to achieve a goal.

- **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
- **Software Agent:**

Software Agents

- Sometimes, the environment may not be the real world
  - E.g., flight simulator, video games, Internet
  - They are all artificial but very complex environments
  - Those agents working in these environments are called
- Software agent (softbots)
- Because all parts of the agent are software

Agent and Environment Relationship

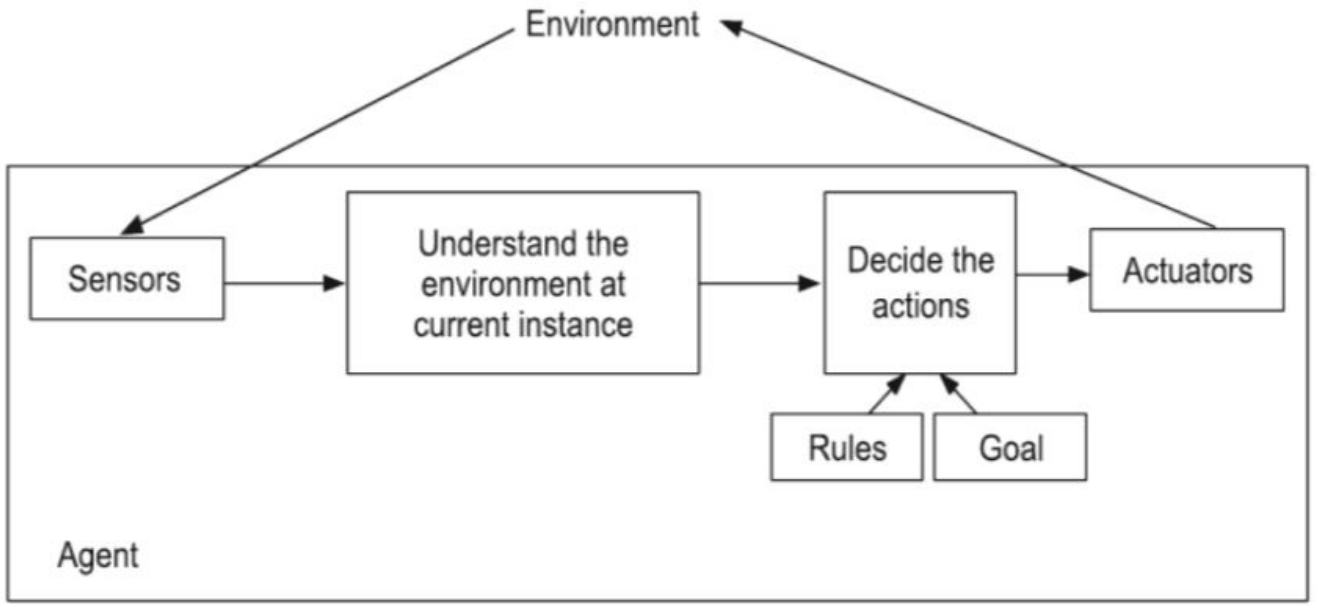


Figure 5.2 Agent-environment relationship.

Agents and Environment

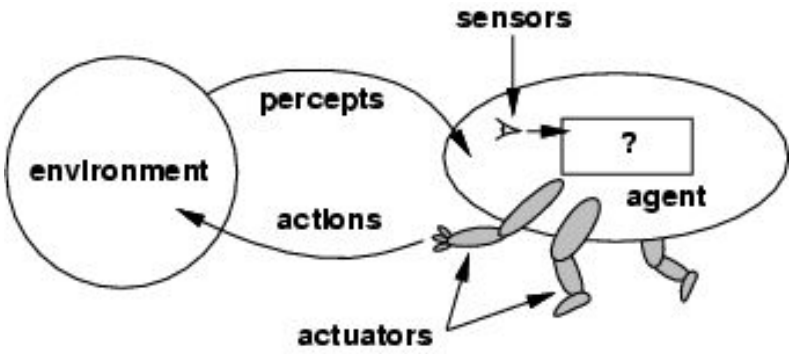
Percept

- Agent's perceptual inputs at any given instant

Percept sequence

- Complete history of everything that the agent has ever perceived

Agents and Environment



- 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

Representation of Agent Function as a Subset of Agent Program

Example: Auto-door opening and closing system.

Two cameras, one inside and other outside, cover this area.

A simple agent function is that if any person is standing in that area, open the door or else close.

So, a function is like

- If area is empty, then close the door.
- If area is occupied, then open the door.

An agent program uses this function to achieve this complete expected behaviour. The program takes care of following activities:

- Continuous scanning of this area
- Opening or closing of the door or keeping it in the same position based on the output of function
- Handling all combinations and possible issues
- Handling the exceptions to ensure the smooth functioning
- Ultimately achieving the results in all possible scenarios

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Rationality

- **Rationality**. **Rationality** is nothing but status of being reasonable, sensible, and having good sense of judgment.
- **Rationality** is concerned with expected actions and results depending upon what the agent has perceived.
- Rationality is a normative concept that stands for acting based on logical reasoning.
- Performing actions with the aim of obtaining useful information is an important part of **rationality**.

What is a Rational Agent?

- A rational agent is an agent that behaves logically and does the right things.
- In artificial intelligence and even in other disciplines like economics, game theory, decision theory, a rational agent is an agent that chooses to perform an action which leads to an expected optimal result.
- Along with all sensors and actuators, the agent is provided with complete specifications of the problem and the task to be performed.
- Based on this information, the agent performs the most logical actions.
- Rational actions are those which can make an agent the most successful.
- A rational agent provides or makes rational rather logical decisions.
- **Typical examples of rational agent : A** person, governing body, decision authority, firm, machine or software.

What is Ideal Rational Agent?

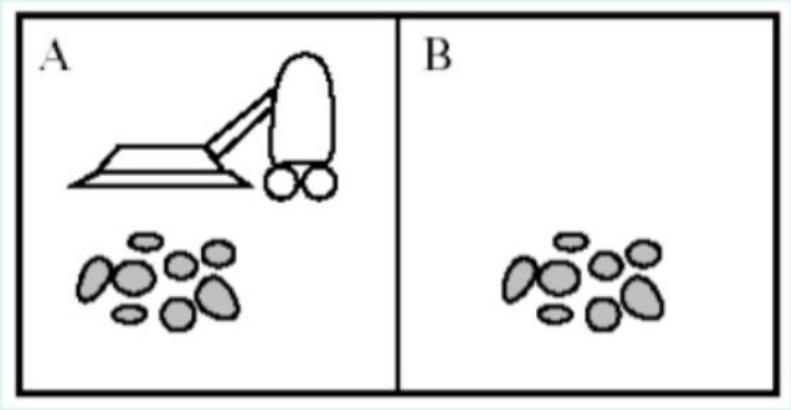
An ideal rational agent is the one, which is capable of doing expected actions to maximize its performance measure, on the basis of −

- Its percept sequence
- Its built-in knowledge base

Rationality of an agent depends on the following −

- The **performance measures**, which determine the degree of success.
- Agent's **Percept Sequence** till now.
- The agent's **prior knowledge about the environment**.
- The **actions** that the agent can carry out.

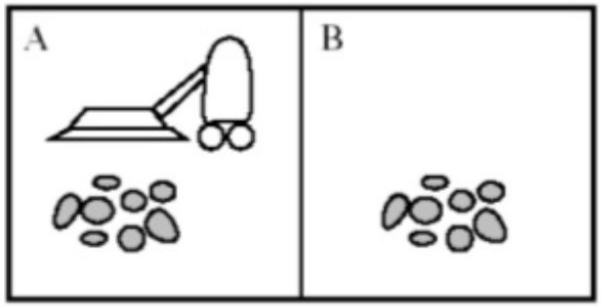
- Environment: squares *A* and *B*
- Percepts: [*location, status*], e.g. [*A, Dirty*]
- Actions: *left*, *right*, *suck*, and *no-op*



- An *agent function* maps percept sequences to actions.

$$f : \text{seq}(P) \rightarrow A$$

- Agent function for vacuum Cleaner example:



Percept sequence	Action
[A,Clean]	*Right*
[A, Dirty]	*Suck*
[B, Clean]	*Left*
[B, Dirty]	*Suck*
[A, Clean],[B, Clean]	*Left*
[A, Clean],[B, Dirty]	*Suck*
…	…

❑ **To design a rational agent we must specify its task environment:**

Performance Measures	used to evaluate how well an agent solves the task at hand
Environment	surroundings beyond the control of the agent
Actuators	used by the agent to perform actions
Sensors	provide information about the current state of the environment

A rational agent always performs right action, where the right action means the action that causes the agent to be most successful in the given percept sequence. The problem the agent solves is characterized by **Performance Measure, Environment, Actuators, and Sensors (PEAS).**

- The task of designing an automated taxi driver:
  - **Performance measure:** Safe, fast, legal, comfortable trip, maximize profits
  - **Environment:** Roads, other traffic, pedestrians, customers
  - **Actuators:** Steering wheel, accelerator, brake, signal, horn
  - **Sensors:** Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard

Rational Agents

- What is rational depends on four things:
  - ***Performance measure***
  - ***Percept sequence: everything agent has seen so far***
  - ***Knowledge agent has about environment***
  - ***Actions agent is capable of performing***
- Ideal Rational Agent
  - Does whatever action is expected to maximize its performance measure, based on percept sequence and built-in knowledge

Ideal Mapping

- Ideal mapping describes behavior of ideal agents

The Mapping Table

- In most cases, mapping is explosively too large to write down explicitly
- In some cases, mapping can be defined via a specification
  - Example: agent to sort a list of numbers
  - Sample table for such an agent
  - Lisp code

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI
- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems
- Problem types and characteristics
- Problem space and search
- Intelligent agent
- Rationality and Rational agent with performance measures
- Flexibility and Intelligent agents
- Task environment and its properties

- Types of agents
- Other aspects of agents
- Constraint satisfaction problems(CSP)
- Crypto arithmetic puzzles
- CSP as a search problem-constrains and representation
- CSP-Backtracking, Role of heuristic
- CSP-Forward checking and constraint propagation
- CSP-Intelligent backtracking

Real-world Problems

Example of rational action performed by any intelligent agent:

Automated Taxi Driver:

Performance Measure: Safe, fast, legal, comfortable trip, maximize profits.

Environment: Roads, other traffic, customers.

Actuators: Steering wheel, accelerator, brake, signal, horn.

Sensors: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard.

Concept of Rationality

- Rational agent
  - One that does the right thing
  - = every entry in the table for the agent function is correct (rational).
- What is correct?
  - The actions that cause the agent to be most successful
  - So we need ways to measure success.

Performance measure

- Performance measure
  - An objective function that determines
    - How the agent does successfully
    - E.g., 90% or 30% ?
- An agent, based on its percepts
  - □ action sequence :

  if desirable, it is said to be performing well.
  - No universal performance measure for all agents

Performance measure

- A general rule:
  - Design performance measures according to
    - What one actually wants in the environment
    - Rather than how one thinks the agent should behave
- E.g., in vacuum-cleaner world
  - We want the floor clean, no matter how the agent behave
  - We don't restrict how the agent behaves

Example of a rational agent

- Performance measure
  - Awards one point for each clean square
    - at each time step, over 10000 time steps
- Prior knowledge about the environment
  - The geography of the environment
  - Only two squares
  - The *effect* of the actions

Example of a rational agent

- Actions that can perform
  - Left, Right, Suck and NoOp
- Percept sequences
  - Where is the agent?
  - Whether the location contains dirt?

- Under this circumstance, the agent is rational.

Omniscience

- An omniscient agent
  - Knows the *actual* outcome of its actions in advance
  - No other possible outcomes
  - However, impossible in real world
- An example
  - crossing a street but died of the fallen cargo door from 33,000ft irrational?

Omniscience

- Based on the circumstance, it is rational.
- As rationality maximizes
  - *Expected* performance
- Perfection maximizes
  - *Actual* performance
- Hence rational agents are not *omniscient*.

Learning

- Does a rational agent depend on only current percept?
  - No, the past percept sequence should also be used
  - This is called **learning**
  - After experiencing an episode, the agent
    - should adjust its behaviors to perform better for the same job next time.

Autonomy

- If an agent just relies on the prior knowledge of its designer rather than its own percepts then the agent lacks ***autonomy***

A rational agent should be autonomous- it should learn what it can to compensate for partial or incorrect prior knowledge.

- E.g., a clock
  - No input (percepts)
  - Run only but its own algorithm (prior knowledge)
  - No learning, no experience, etc.

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Intelligent Agents

- What is an agent ?
  - An **agent** is anything that **perceiving** its environment through **sensors** and **acting** upon that environment through **actuators**
  - Example:
    - Human is an agent
    - A robot is also an agent with cameras and motors
    - A thermostat detecting room temperature.

Intelligent Agents

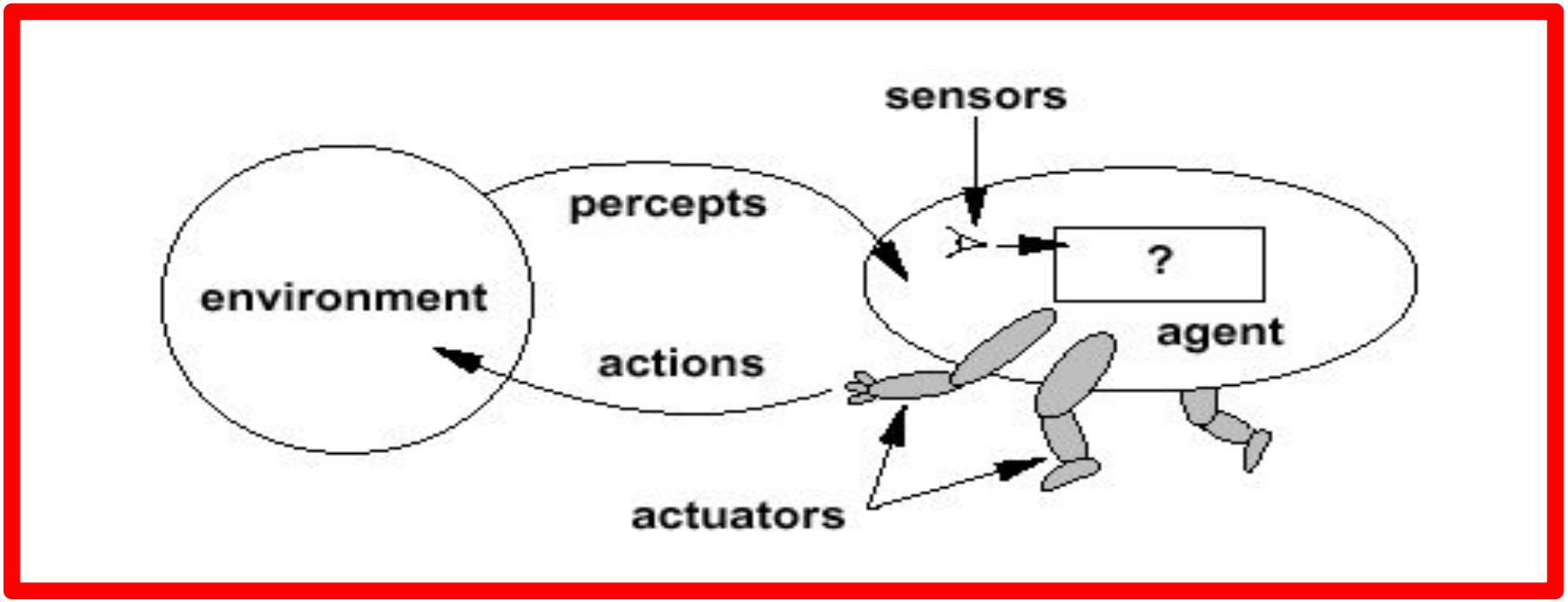
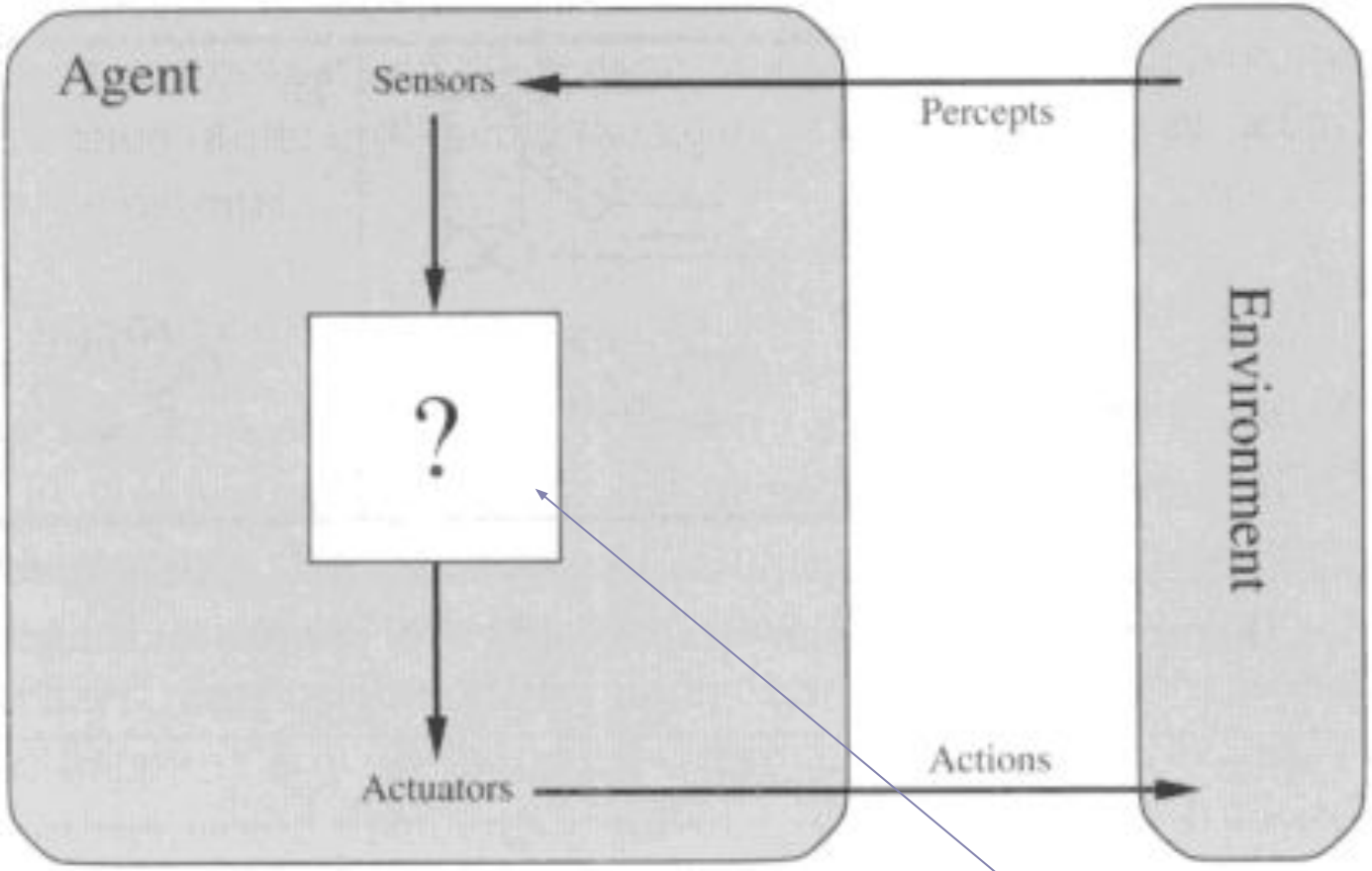


Diagram of an agent



Simple Terms

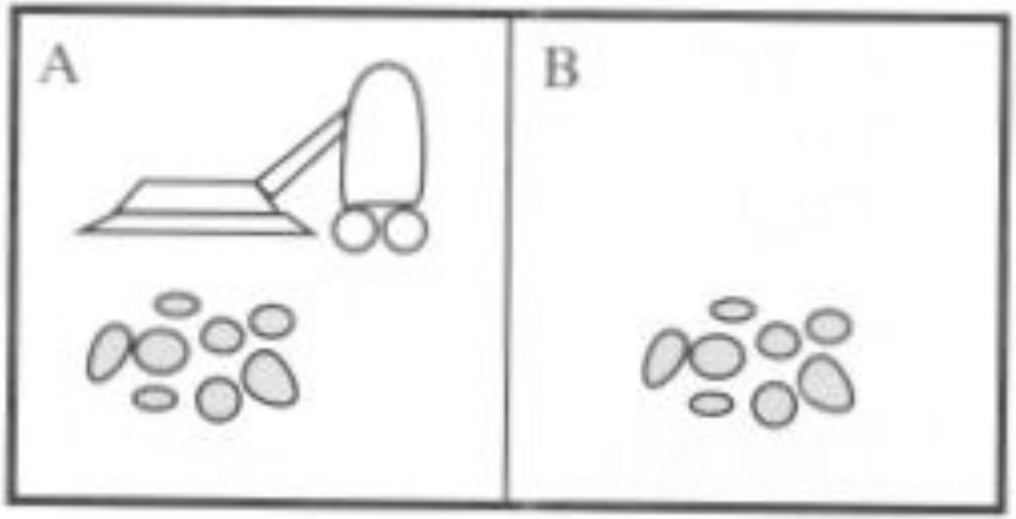
- Percept
  - Agent's perceptual inputs at any given instant
- Percept sequence
  - Complete history of everything that the agent has ever perceived.

Agent function & program

- Agent's behavior is *mathematically* described by
  - **Agent function**
  - A function mapping any given percept sequence to an action
- *Practically* it is described by
  - An **agent program**
  - The real implementation

Vacuum-cleaner world

- Perception: Clean or Dirty? where it is in?
- Actions: Move left, Move right, suck, do nothing



Program implements the agent function tabulated in Fig.

Function Reflex-Vacuum-Agent([*location,status*]) return an action

If *status = Dirty* **then return** *Suck*

else if *location = A* **then return** *Right*

else if *location = B* **then return** *left*

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*
⋮	⋮
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Clean*]	*Right*
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Dirty*]	*Suck*
⋮	⋮

Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Task environments

- Task environments are the problems
  - While the rational agents are the solutions
- Specifying the task environment
  - PEAS description as fully as possible
    - Performance
    - Environment
    - Actuators
    - Sensors

    In designing an agent, the first step must always be to specify the task environment as fully as possible.
- Use automated taxi driver as an example

Task environments

- Performance measure
  - How can we judge the automated driver?
  - Which factors are considered?
    - getting to the correct destination
    - minimizing fuel consumption
    - minimizing the trip time and/or cost
    - minimizing the violations of traffic laws
    - maximizing the safety and comfort, etc.

Task environments

- Environment
  - A taxi must deal with a variety of roads
  - Traffic lights, other vehicles, pedestrians, stray animals, road works, police cars, etc.
  - Interact with the customer

Task environments

- Actuators (for outputs)
  - Control over the accelerator, steering, gear shifting and braking
  - A display to communicate with the customers
- Sensors (for inputs)
  - Detect other vehicles, road situations
  - GPS (Global Positioning System) to know where the taxi is
  - Many more devices are necessary

Task environments

- A sketch of automated taxi driver

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard

Figure 2.4 PEAS description of the task environment for an automated taxi.

Properties of task environments

- Fully observable vs. Partially observable
  - If an agent's sensors give it access to the complete state of the environment at each point in time then the environment is <u>effectively and fully observable</u>
    - if the sensors detect all aspects
    - That are relevant to the choice of action

Properties of task environments

- Partially observable

An environment might be Partially observable because of noisy and inaccurate sensors or because parts of the state are simply missing from the sensor data.

Example:

- A local dirt sensor of the cleaner cannot tell
- Whether other squares are clean or not

Properties of task environments

- Deterministic vs. stochastic
  - next state of the environment Completely determined by the current state and the actions executed by the agent, then the environment is deterministic, otherwise, it is Stochastic.
  - Strategic environment: deterministic except for actions of other agents

-Cleaner and taxi driver are:

- Stochastic because of some unobservable aspects noise or unknown

Properties of task environments

- Episodic vs. sequential
  - An episode = agent's single pair of perception & action
  - The quality of the agent's action does not depend on other episodes

- Every episode is independent of each other
  - Episodic environment is simpler

- The agent does not need to think ahead

- Sequential
  - Current action may affect all future decisions

  -Ex. Taxi driving and chess.

Properties of task environments

- Static vs. dynamic
  - A dynamic environment is always changing over time
- E.g., the number of people in the street
  - While static environment
- E.g., the destination
- Semidynamic
  - environment is not changed over time
  - but the agent's performance score does

Properties of task environments

- Discrete vs. continuous
  - If there are a limited number of distinct states, clearly defined percepts and actions, the environment is discrete
  - E.g., Chess game
  - Continuous: Taxi driving

Properties of task environments

- Single agent VS. multiagent
  - Playing a crossword puzzle – single agent
  - Chess playing – two agents
  - Competitive multiagent environment
- Chess playing
  - Cooperative multiagent environment
- Automated taxi driver
- Avoiding collision

Properties of task environments

- Known vs. unknown

This distinction refers not to the environment itslef but to the agent's (or designer's) state of knowledge about the environment.

-In known environment, the outcomes for all actions are given. (example: solitaire card games).

- If the environment is unknown, the agent will have to learn how it works in order to make good decisions.(example: new video game).

Examples of task environments

Task Environment	Observable	Deterministic	Episodic	Static	Discrete	Agents
Crossword puzzle	Fully	Deterministic	Sequential	Static	Discrete	Single
Chess with a clock	Fully	Strategic	Sequential	Semi	Discrete	Multi
Poker	Partially	Strategic	Sequential	Static	Discrete	Multi
Backgammon	Fully	Stochastic	Sequential	Static	Discrete	Multi
Taxi driving	Partially	Stochastic	Sequential	Dynamic	Continuous	Multi
Medical diagnosis	Partially	Stochastic	Sequential	Dynamic	Continuous	Single
Image-analysis	Fully	Deterministic	Episodic	Semi	Continuous	Single
Part-picking robot	Partially	Stochastic	Episodic	Dynamic	Continuous	Single
Refinery controller	Partially	Stochastic	Sequential	Dynamic	Continuous	Single
Interactive English tutor	Partially	Stochastic	Sequential	Dynamic	Discrete	Multi

Figure 2.6 Examples of task environments and their characteristics.

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Structure of agents

- Agent = architecture + program
  - Architecture = some sort of computing device (sensors + actuators)
  - (Agent) Program = some function that implements the agent mapping = “?”
  - Agent Program = Job of AI

Agent programs

- Input for Agent Program
  - Only the current percept
- Input for Agent Function
  - The entire percept sequence
  - The agent must remember all of them
- Implement the agent program as
  - A look up table (agent function)

Agent programs

- Skeleton design of an agent program

```
function TABLE-DRIVEN-AGENT(percept) returns action
  static: percepts, a sequence, initially empty
          table, a table, indexed by percept sequences, initially fully specified

  append percept to the end of percepts
  action ← LOOKUP(percepts, table)
  return action
```

Agent programs

- P = the set of possible percepts
- T= lifetime of the agent
  - The total number of percepts it receives
- Size of the look up table $\sum_{t=1}^{T} |P|^t$
- Consider playing chess
  - P =10, T=150
  - Will require a table of at least 10^{150} entries

Agent programs

- Despite of huge size, look up table does what we want.
- The key challenge of AI
  - Find out how to write programs that, to the extent possible, produce rational behavior
    - From a small amount of code
    - Rather than a large amount of table entries
  - E.g., a five-line program of Newton's Method
  - V.s. huge tables of square roots, sine, cosine, …

Types of agent programs

- Four types
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based agents

Simple reflex agents

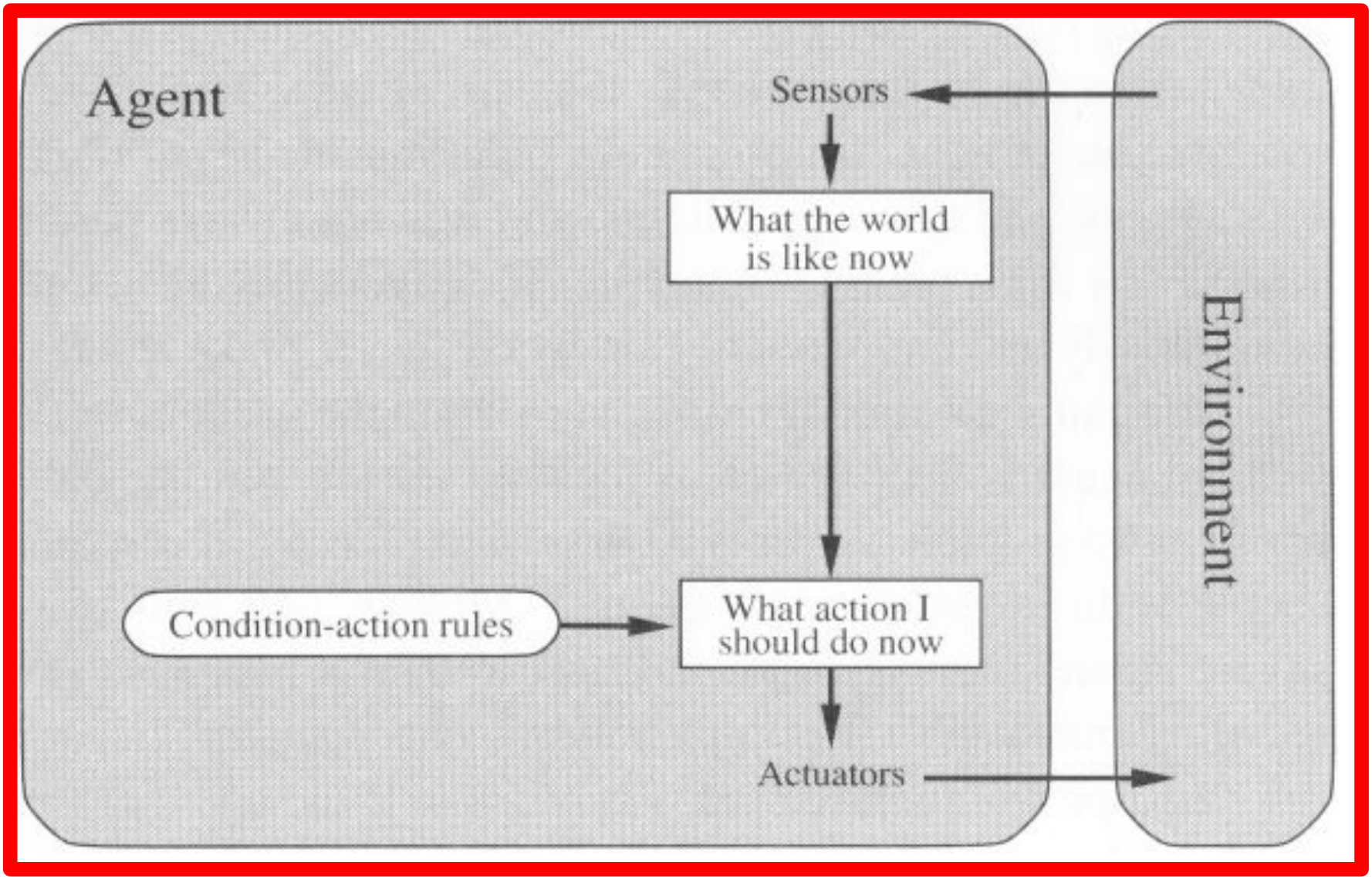
- It uses just ***condition-action rules***
  - The rules are like the form "if … then …"
  - efficient but have narrow range of applicability
  - Because knowledge sometimes cannot be stated explicitly
  - Work only
    - if the environment is fully observable

Simple reflex agents

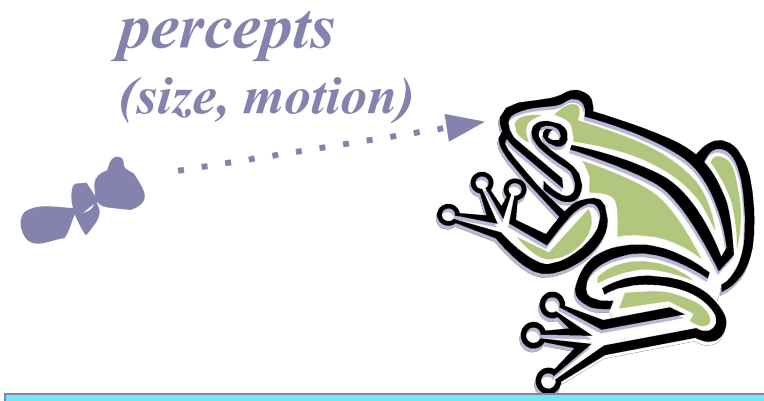

```
function SIMPLE-REFLEX-AGENT(percept) returns action
    static: rules, a set of condition-action rules

    state ← INTERPRET-INPUT(percept)
    rule ← RULE-MATCH(state, rules)
    action ← RULE-ACTION[rule]
    return action
```

Simple reflex agents



A Simple Reflex Agent in Nature



RULES:

(1) If small moving object,
then activate SNAP

(2) If large moving object,
then activate AVOID and inhibit SNAP

ELSE (not moving) then NOOP

needed for completeness

Action: SNAP or AVOID or NOOP

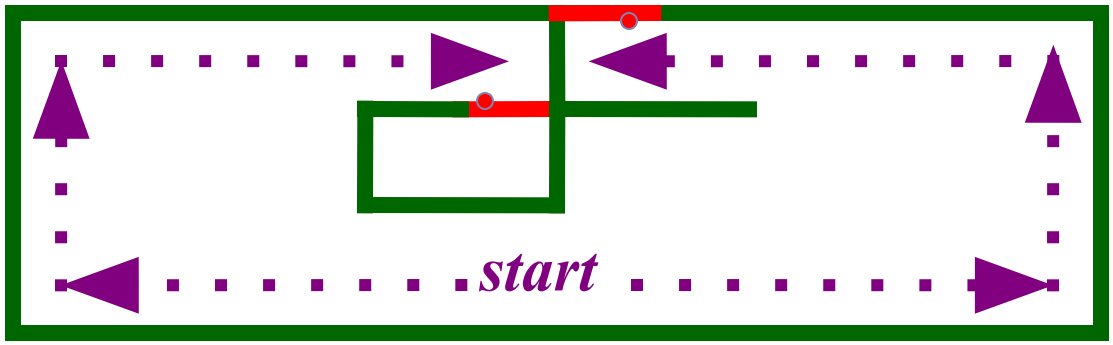
Model-based Reflex Agents

- For the world that is partially observable
  - the agent has to keep track of an internal state
    - That depends on the percept history
    - Reflecting some of the unobserved aspects
    - E.g., driving a car and changing lane
- Requiring two types of knowledge
  - How the world evolves independently of the agent
  - How the agent's actions affect the world

Example Table Agent With Internal State

IF	THEN
Saw an object ahead, and turned right, and it's now clear ahead	Go straight
Saw an object Ahead, turned right, and object ahead again	Halt
See no objects ahead	Go straight
See an object ahead	Turn randomly

Example Reflex Agent With Internal State: **Wall-Following**



Actions: left, right, straight, open-door

Rules:

1. If open(left) & open(right) and open(straight) then choose randomly between right and left
2. If wall(left) and open(right) and open(straight) then straight
3. If wall(right) and open(left) and open(straight) then straight
4. If wall(right) and open(left) and wall(straight) then left
5. If wall(left) and open(right) and wall(straight) then right
6. If wall(left) and door(right) and wall(straight) then open-door
7. If wall(right) and wall(left) and open(straight) then straight.
8. (Default) Move randomly

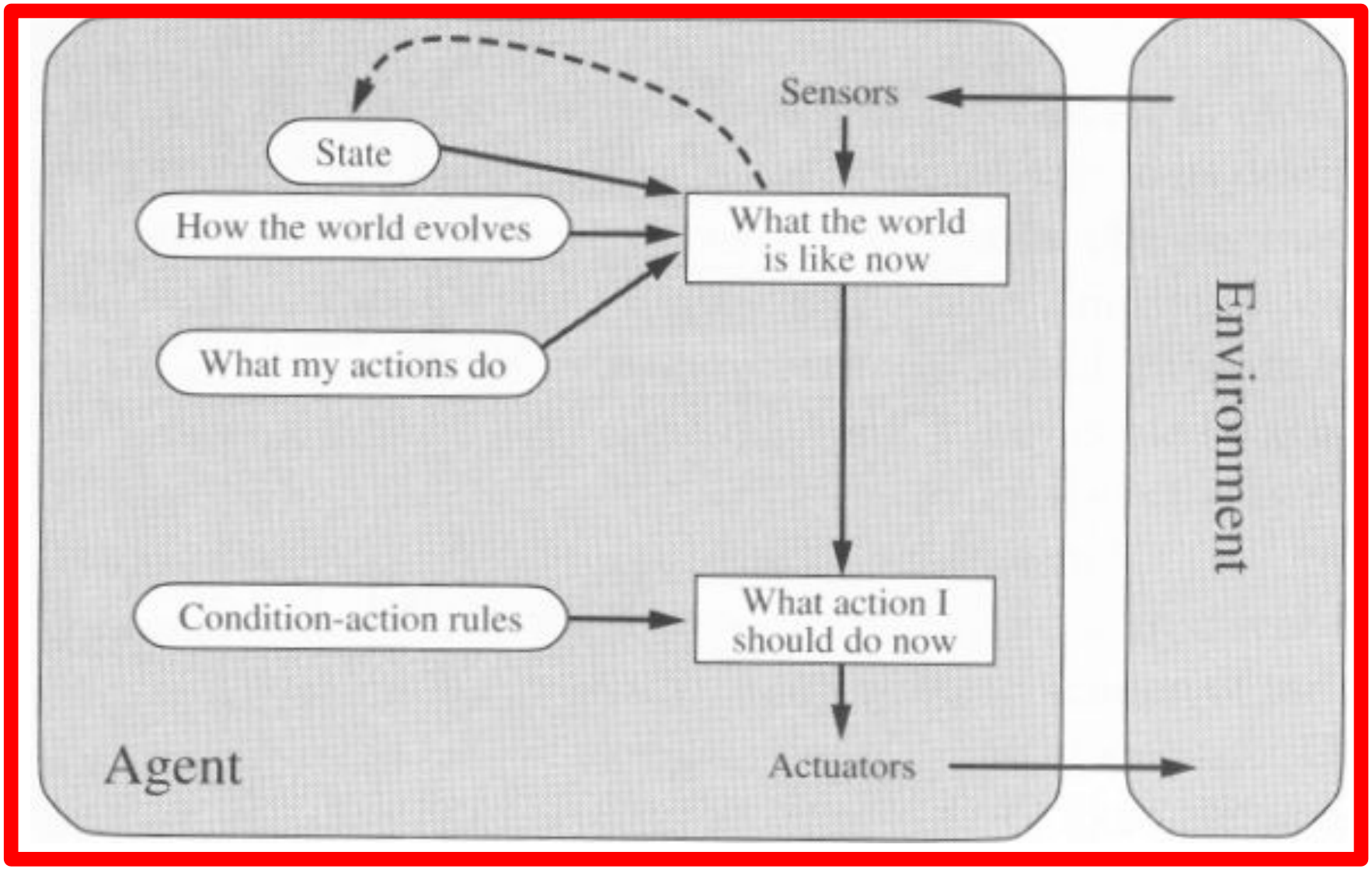
Model-based Reflex Agents


```
function REFLEX-AGENT-WITH-STATE(percept) returns action
  static: state, a description of the current world state
          rules, a set of condition-action rules

  state ← UPDATE-STATE(state, percept)
  rule ← RULE-MATCH(state, rules)
  action ← RULE-ACTION[rule]
  state ← UPDATE-STATE(state, action)
  return action
```

The agent is with memory

Model-based Reflex Agents



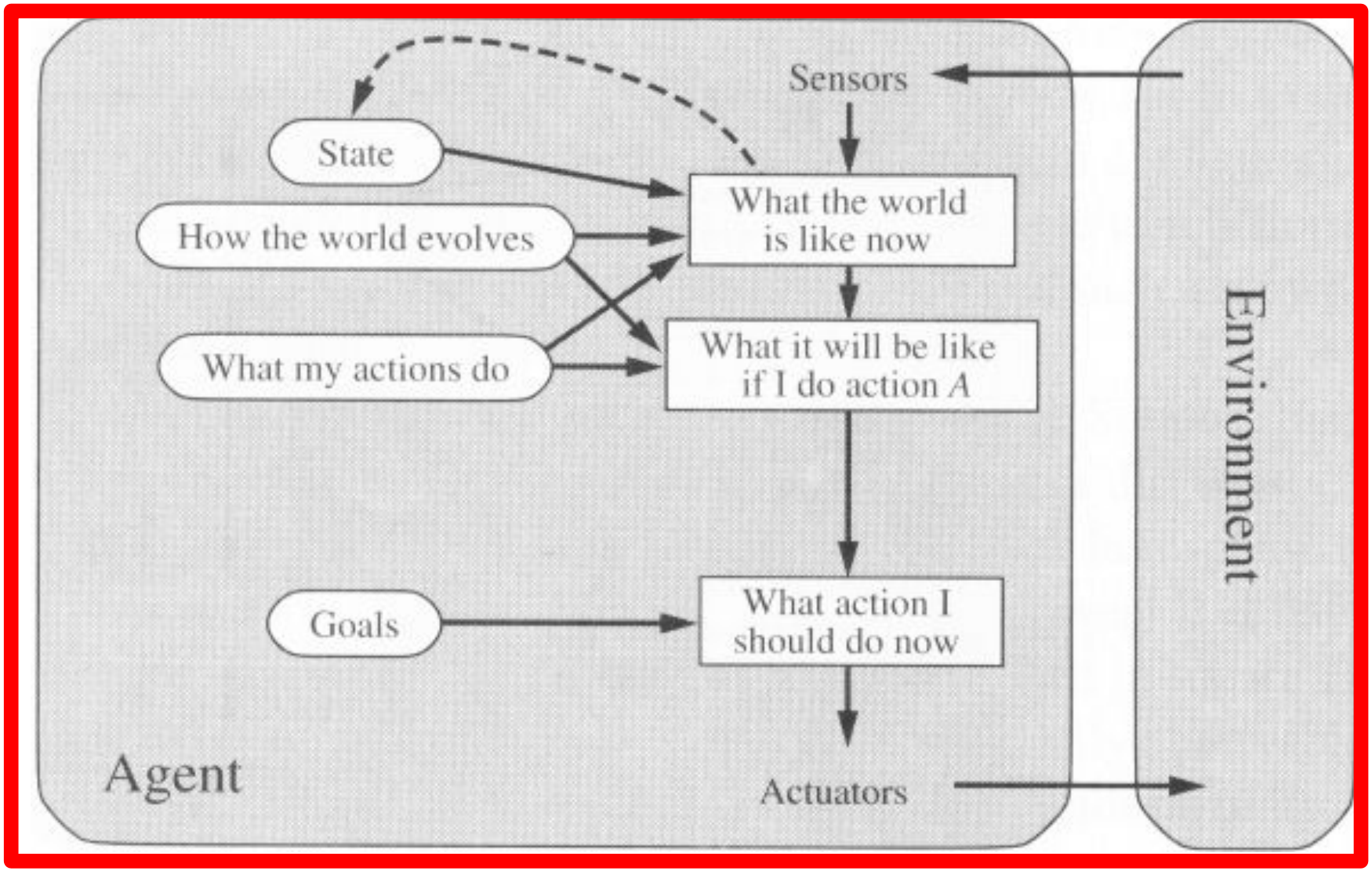
Goal-based agents

- Current state of the environment is always not enough
- The goal is another issue to achieve
  - Judgment of rationality / correctness
- Actions chosen goals, based on
  - the current state
  - the current percept

Goal-based agents

- Conclusion
  - Goal-based agents are less efficient
  - but more flexible
    - Agent Different goals different tasks
  - Search and planning
    - two other sub-fields in AI
    - to find out the action sequences to achieve its goal

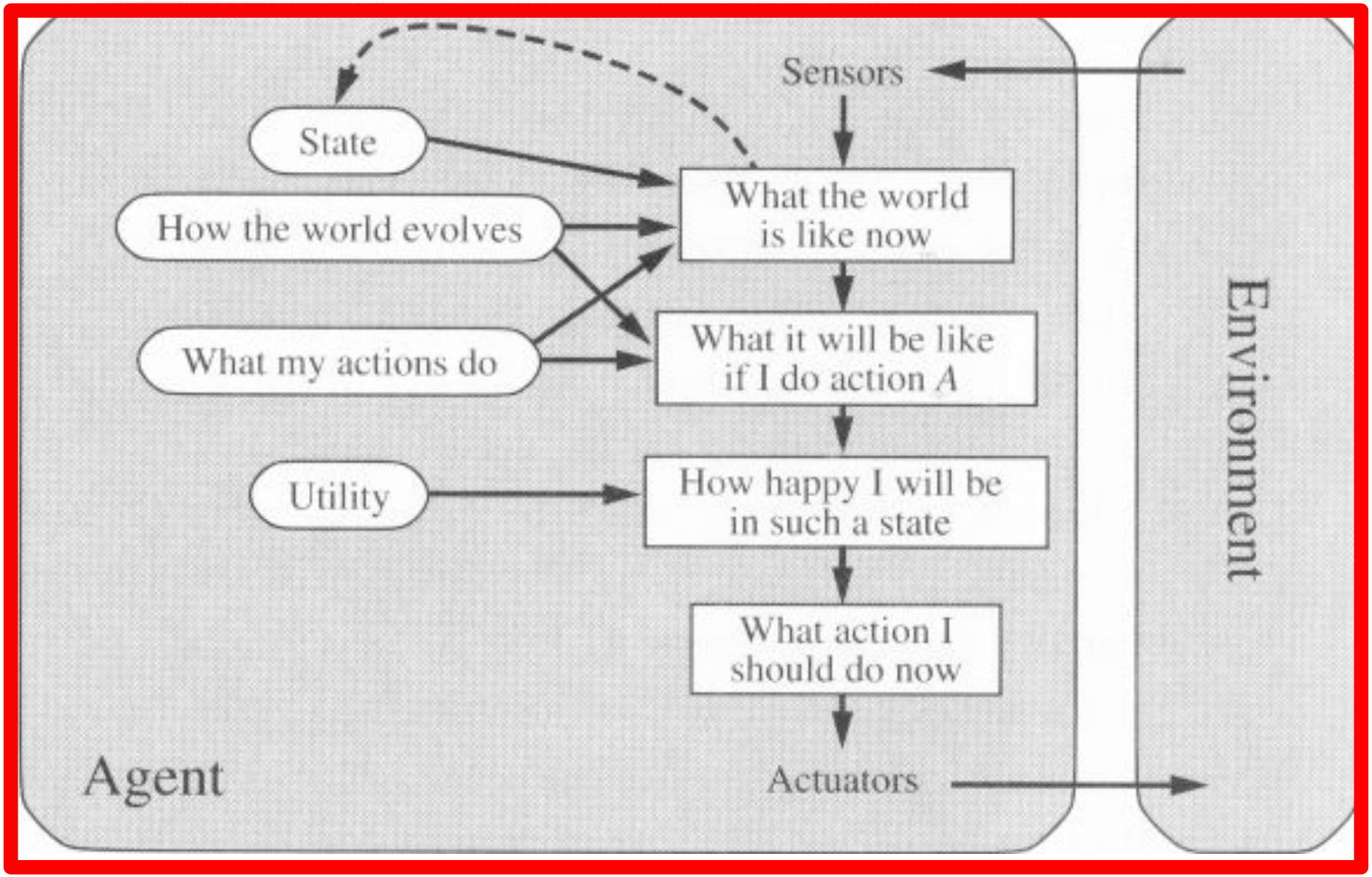
Goal-based agents



Utility-based agents

- Goals alone are not enough
  - to generate **high-quality** behavior
  - E.g. meals in Canteen, good or not ?
- Many action sequences the goals
  - some are better and some worse
  - If goal means success,
  - then **utility** means the degree of success (how successful it is)

Utility-based agents(4)



Utility-based agents

- it is said state A has higher utility
  - If state A is more preferred than others
- Utility is therefore a function
  - that maps a state onto a real number
  - the degree of success

Utility-based agents (3)

- Utility has several advantages:
  - When there are conflicting goals,
    - Only some of the goals but not all can be achieved
    - utility describes the appropriate trade-off
  - When there are several goals
    - None of them are achieved **certainly**
    - utility provides a way for the decision-making

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Learning Agents

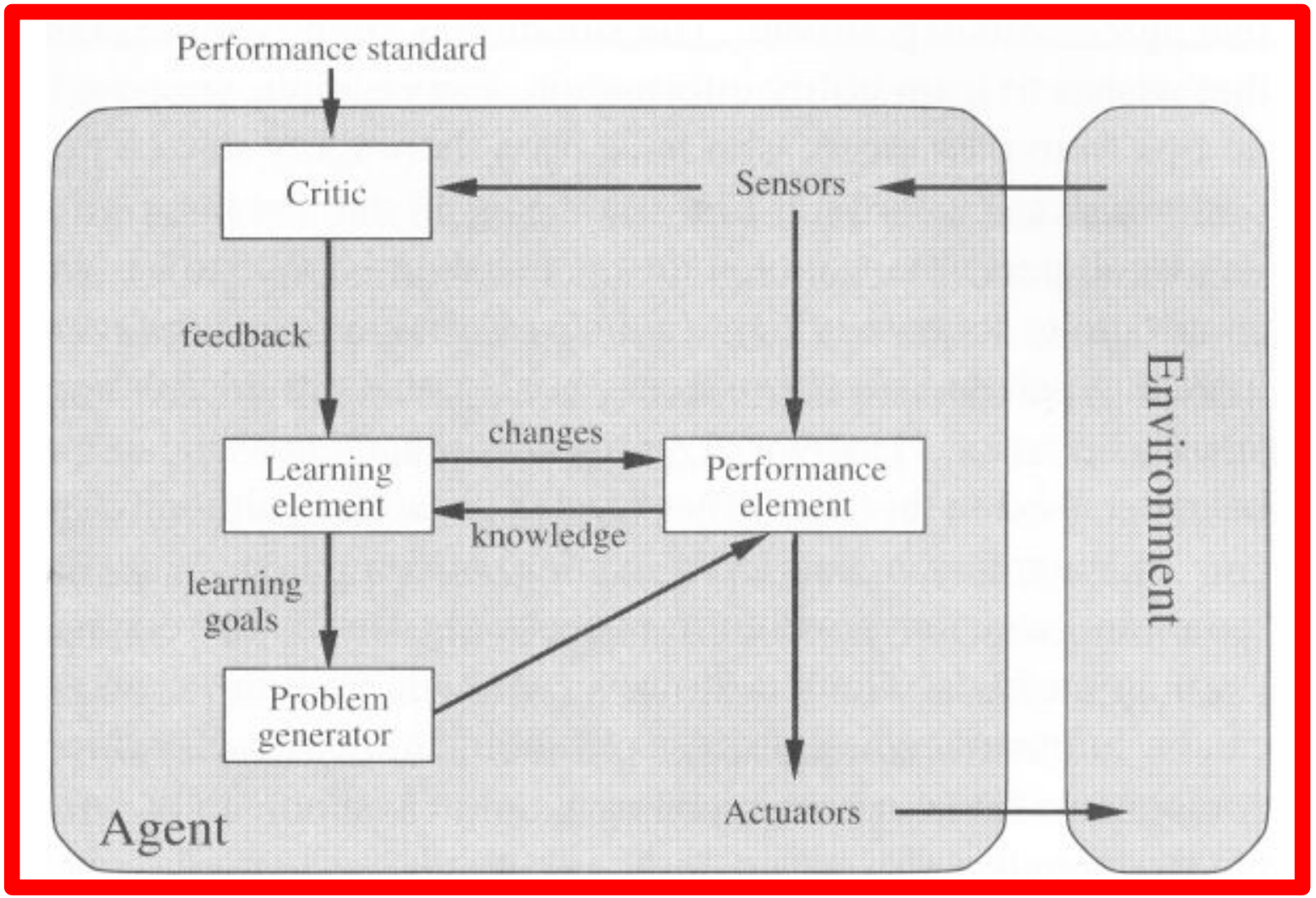
- After an agent is programmed, can it work immediately?
  - No, it still need teaching
- In AI,
  - Once an agent is done
  - We teach it by giving it a set of examples
  - Test it by using another set of examples
- We then say the agent learns
  - A learning agent

Learning Agents

- Four conceptual components
  - Learning element
    - Making improvement
  - Performance element
    - Selecting external actions
  - Critic
    - Tells the Learning element how well the agent is doing with respect to fixed performance standard.

    (Feedback from user or examples, good or not?)
  - Problem generator
    - Suggest actions that will lead to new and informative experiences.

Learning Agents



Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Constraint satisfaction problems (CSPs)

- CSP:
  - state is defined by variables X_i with values from domain D_i
  - goal test is a set of constraints specifying allowable combinations of values for subsets of variables
- Allows useful general-purpose algorithms with more power than standard search algorithms

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

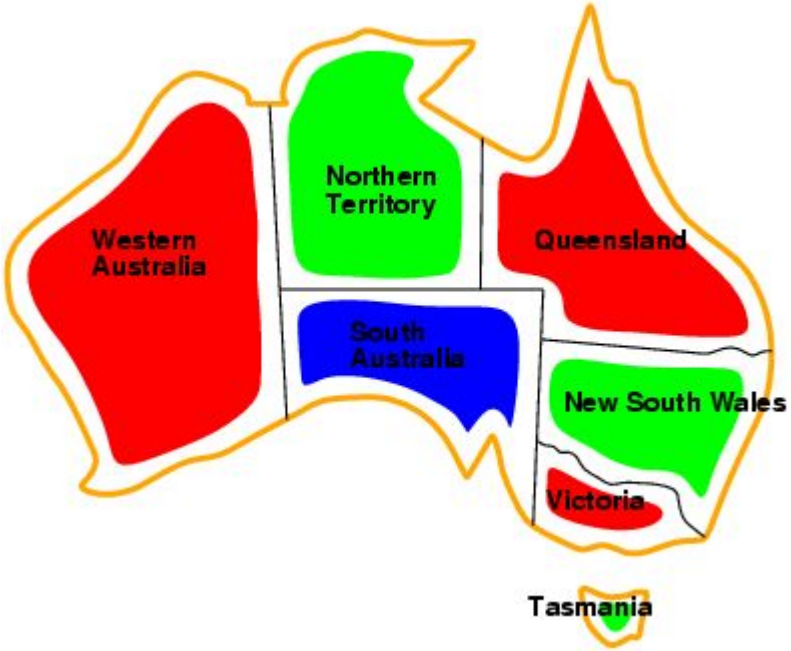
•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Example: Map-Coloring

- Variables *WA, NT, Q, NSW, V, SA, T*
- Domains D_i = {red,green,blue}
- Constraints: adjacent regions must have different colors
- e.g., WA ≠ NT

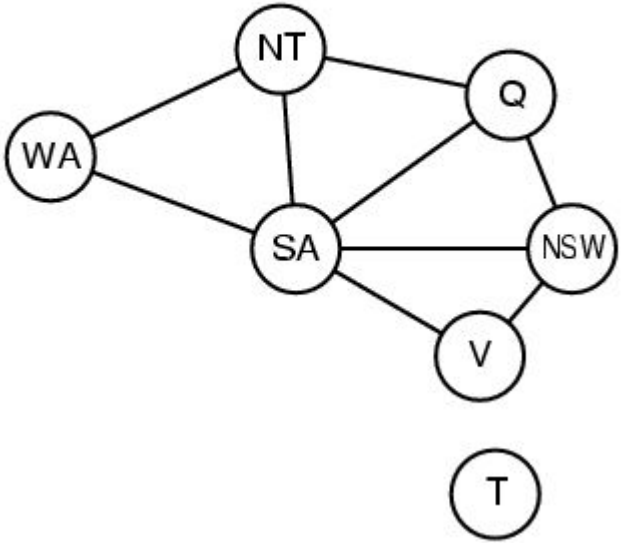
Example: Map-Coloring



- Solutions are complete and consistent assignments, e.g., WA = red, NT = green,Q = red,NSW = green,V = red,SA = blue,T = green

Constraint graph

- Binary CSP: each constraint relates two variables
- Constraint graph: nodes are variables, arcs are constraints



Varieties of CSPs

- Discrete variables
  - finite domains:
    - n variables, domain size d ⍰ $O(d^n)$ complete assignments
    - e.g., 3-SAT (NP-complete)
  - infinite domains:
    - integers, strings, etc.
    - e.g., job scheduling, variables are start/end days for each job
    - need a constraint language, e.g., $StartJob_1 + 5 \leq StartJob_3$
- Continuous variables
  - e.g., start/end times for Hubble Space Telescope observations
  - linear constraints solvable in polynomial time by linear programming

Varieties of constraints

- Unary constraints involve a single variable,
  - e.g., SA ≠ green
- Binary constraints involve pairs of variables,
  - e.g., SA ≠ WA
- Higher-order constraints involve 3 or more variables,
  - e.g., SA ≠ WA ≠ NT

Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

Cryptarithmetic

- SEND+MORE = MONEY
- Initial State: no variable has a value
- Operators:
  - assign a variable a digit (0..9) (no dups)
  - unassign a variable
- Goal: arithmetic statement is true.
- Example of Constraint Satisfaction Problem

Cryptarithmetic , An example

Rule 1 : 0 to 9
Rule 2: 1 Letter □ no 2 nos
1 no □ no 2 letters
Rule 3: maximum carry over is one

M = Need a carry
Possibility □ 1
M=1

4th column:
O:

10 to 19 possible except number 11. Bz O Will Get value 1 which is already assigned for M.

So possibilities are

10, 12 to 19

NC : No CARRY

	1			NC
	S	E	N	D
	M	O	R	E
1				
M	O	N	E	Y

1

Assume S = 9:

W/O Carry:

S + M = 10 + O

9 + 1 = 10

With Carry:

C+S+ M = 10 + O

1+ 9 + 1 = 10 + O

Assume S = 8:

w/o Carry :

8+ 1 = 10 + O

With Carry:

1+8+1 = 10 + O

1				NC
	S	E	N	D
	M	O	R	E
	1	0		
M	O	N	E	Y
1	0			

3rd Column:

w/o carry:

E + O = N

E + 0 = N (Any number + 0 = same num)

So this not allowed because both N and O

Will get same number)

With Carry:

C+ E + O = N

1+ E + 0 = N

1+ E = N ------ Eq. 1

1		1		NC
	S	E	N	D
	M	O	R	E
	1	0		
M	O	N	E	Y
1	0			

2nd Column:

With Carry:

C+N+R = 10+E

1+N+R = 10+ E ------- **Eq.2**

Sub Eq.1 in eq. 2

1+1+E+R = 10 +E

R = 8

Without Carry:

N+R = 10+E

1+E +R = 10+E

R = 9

Now in 3rd column, lets finalize the value of E.
If E = 9 (highest possible no), then 1+9 = 10 which makes N as '0' . But '0' already allotted for ALPHABET 'O'. So E cannot be 9.

If E = 8 (Next highest possible no), then 1 + 8 = 9 (A single digit no) .

SO For E, any value 8 till 2 (1 and 0 already allotted) is possible. In that case, in 4th column No Carry (NC) will come.

1	NC	1		NC
	S	E	N	D
	M	O	R	E
	1	0	8	
M	O	N	E	Y
1	0			

Coming to 4th column again to find S:

Since no carry for 4th column, S should be 9 ,not 8.

Since S + M = 10 + O, if we put S = 8, we will get

8 + 1 = 10 + 0 [which is only 9 not 10,

So S=9 is fixed]

Now, we will get a two digit number

(i,e) 10 as result and carry will be

forwarded to next (5th) column.

1	NC	1		NC
	S	E	N	D
	9			
	M	O	R	E
	1	0	8	
M	O	N	E	Y
1	0			

1st column:

D + E = 10 + Y

10
11
12 **3+9, 4+8, 5+7**
13 **1+12, 2+11, 3+10, 4+9, 5+8, 6+7**
14
15
16
17
18
19

All other combinations will be ruled out since its not satisfying constraints. So finally we will get
5+7
6 +7

1	NC	1	1	NC
	S	E	N	D
	9	5	6	7
	M	O	R	E
	1	0	8	5
M	O	N	E	Y
1	0	6	5	2

1st column:

D + E = 10 + Y

Two possibilities : 5+7, 6 +7

Common number : 7

Now we want to check, value 7 should be assigned to either D or E.

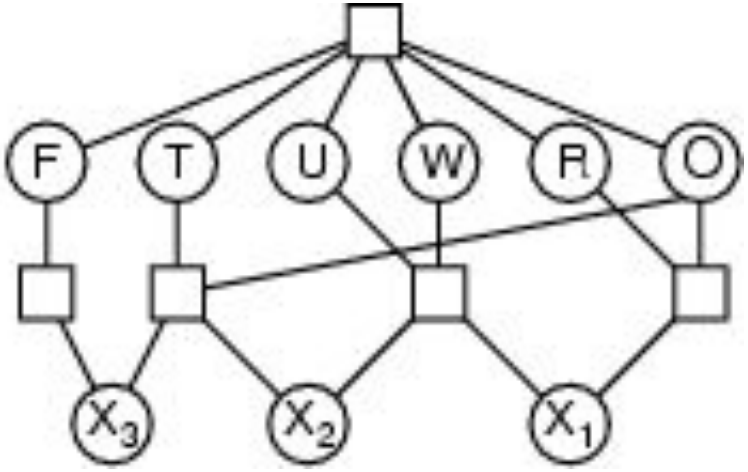
If we take E = 7, in 3rd column we will get
1+7+0 = 8 (But 8 already allotted to R)
So D =7 is fixed.

Now for E, we want to check, whether it can be be assigned with 5 or 6.
If we put E = 6, in 3rd column we will get
1+6+0 = 7 (But 7 already allotted to D)
So E = 5 is fixed.
Now 1st column is filled which gives Y = 2 3rd column Is also filled which gives N = 6.

1	NC	1	1	NC
	S	E	N	D
	9	5	6	7
	M	O	R	E
	1	0	8	5
M	O	N	E	Y
1	0	6	5	2

Example: Cryptarithmetic

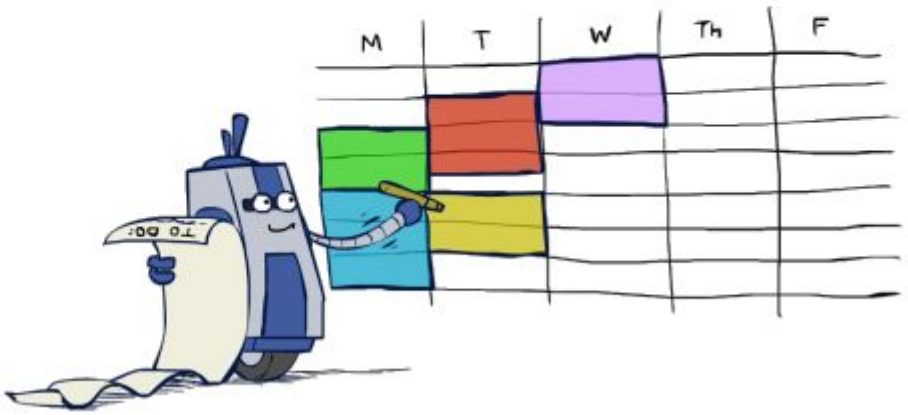
```
   T W O
+  T W O
-------
F O U R
```



- Variables: *F T U W R O* $X_1 X_2 X_3$
- Domains: {*0,1,2,3,4,5,6,7,8,9*} {0,1}
- Constraints: *Alldiff (F,T,U,W,R,O)*
  - $O + O = R + 10 \cdot X_1$
  - $X_1 + W + W = U + 10 \cdot X_2$
  - $X_2 + T + T = O + 10 \cdot X_3$
  - $X_3 = F$, $T \neq 0$, $F \neq 0$

Real-world CSPs

- Scheduling problems: e.g., when can we all meet?
- Timetabling problems: e.g., which class is offered when and where?
- Assignment problems: e.g., who teaches what class
- Hardware configuration
- Transportation scheduling
- Factory scheduling
- Circuit layout
- Fault diagnosis
- … lots more!



Unit 1 List of Topics

- Introduction to AI-AI techniques
- Problem solving with AI

- AI Models, Data acquisition and learning aspects in AI
- Problem solving- Problem solving process, Formulating problems

- Problem types and characteristics
- Problem space and search

- Intelligent agent
- Rationality and Rational agent with performance measures

•Flexibility and Intelligent agents
•Task environment and its properties

•Types of agents
•Other aspects of agents

•Constraint satisfaction problems(CSP)
•Crypto arithmetic puzzles

•CSP as a search problem-constrains and representation
•CSP-Backtracking, Role of heuristic

•CSP-Forward checking and constraint propagation
•CSP-Intelligent backtracking

CSP – As a search problem

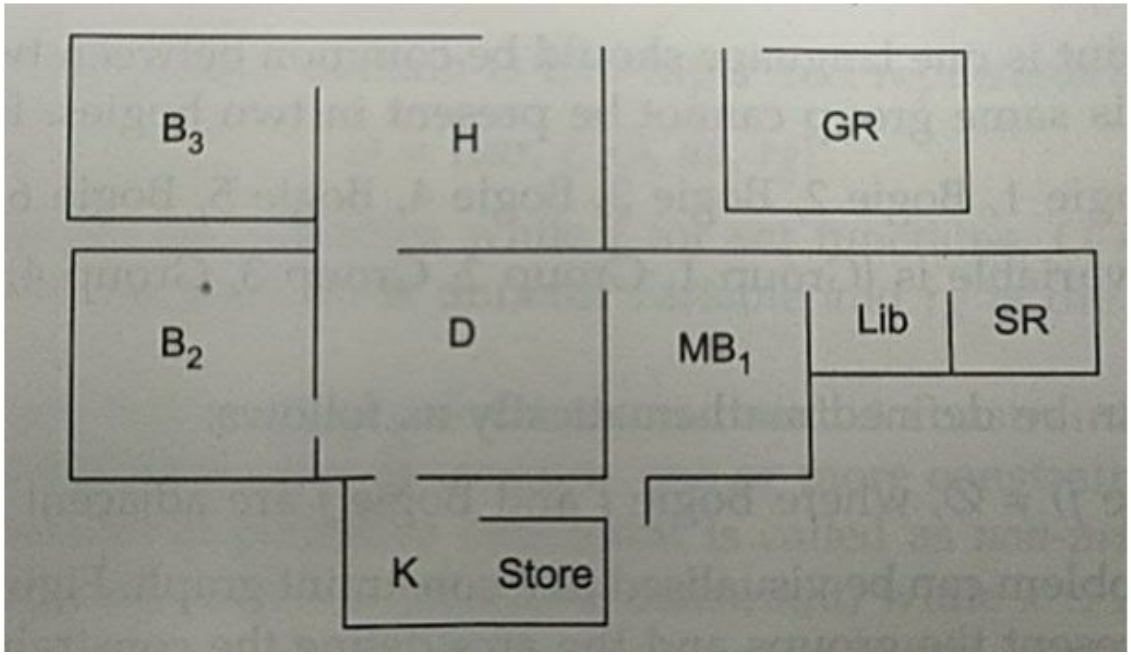
- Initial state
- Successor function
- Goal test
- Path cost

CSP Solution : Complete assignment with no exception

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

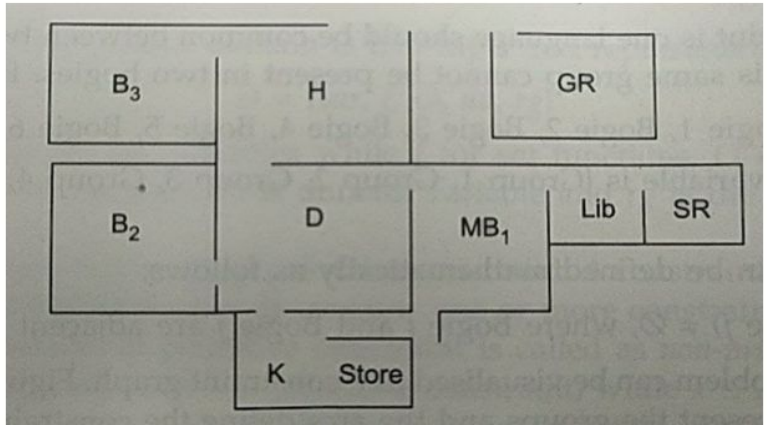
Room Coloring Problem



Representation as a Search problem

- Lines - Adjacency rooms
- Dotted Lines - No Adjacency

Room Coloring Problem



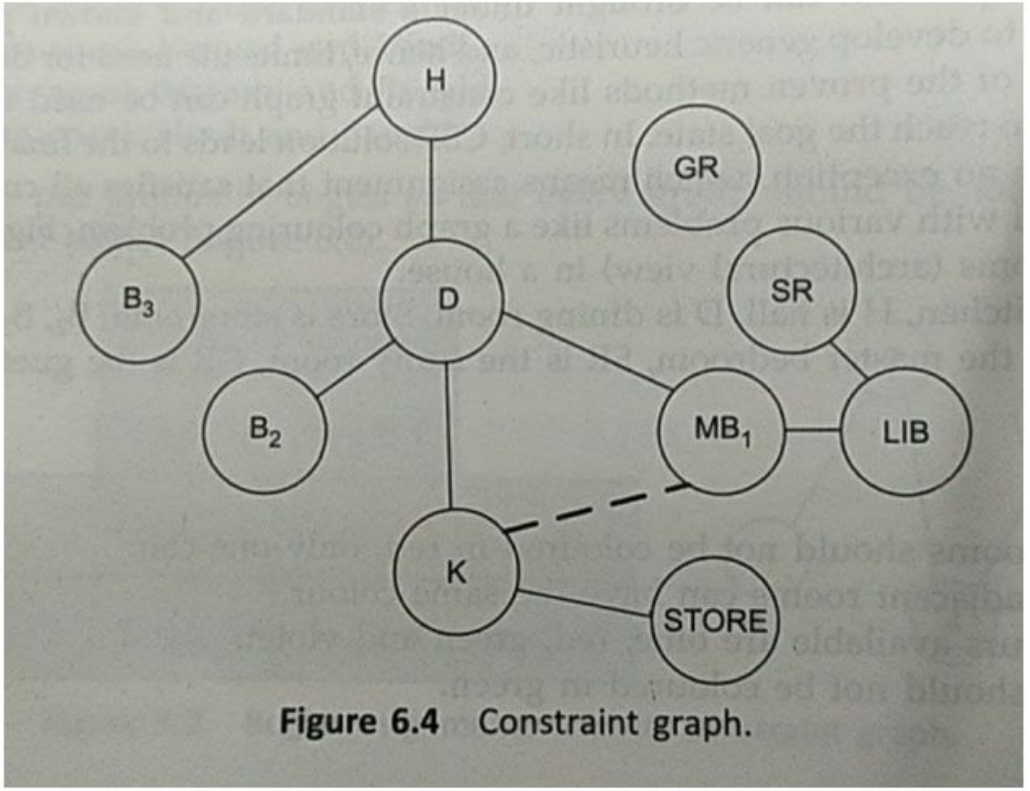


Figure 6.4 Constraint graph.

Search tree generation : Snapshot

- 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

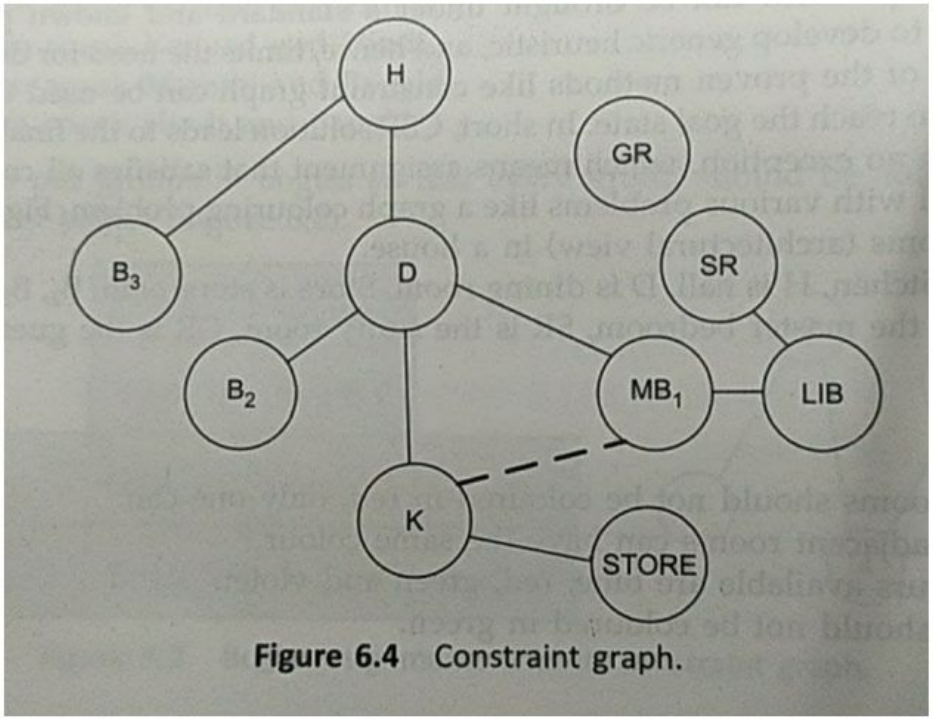


Figure 6.4 Constraint graph.

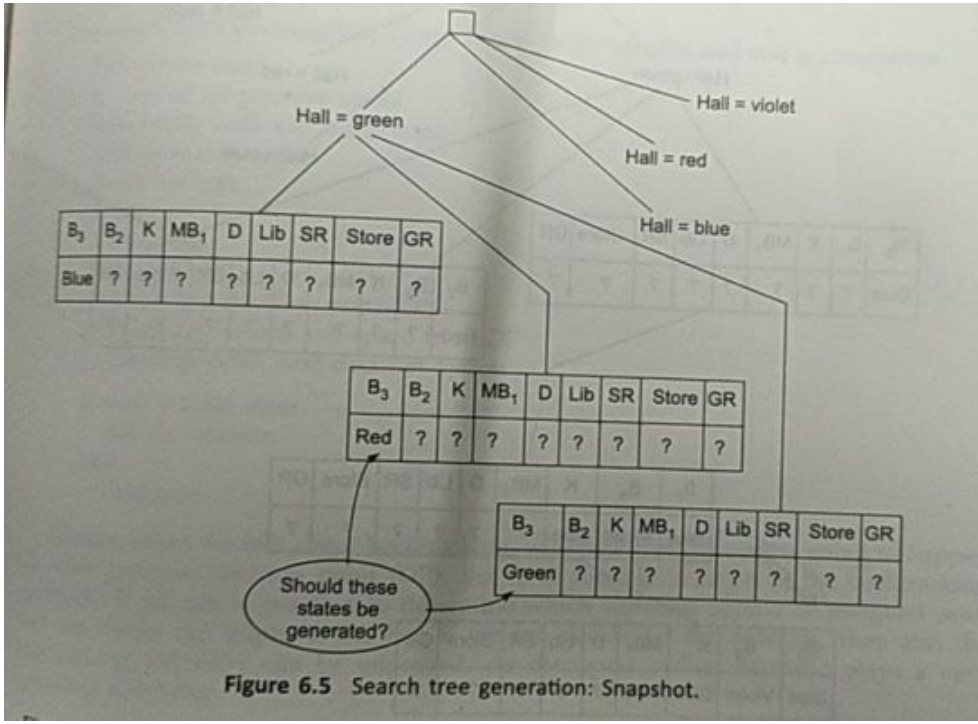


Figure 6.5 Search tree generation: Snapshot.

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

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

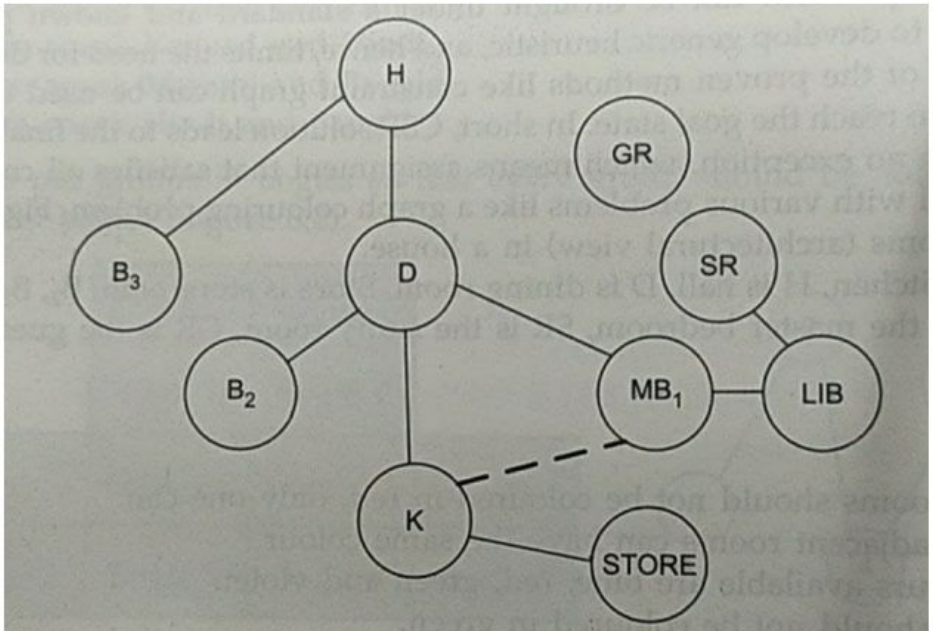


Figure 6.4 Constraint graph.

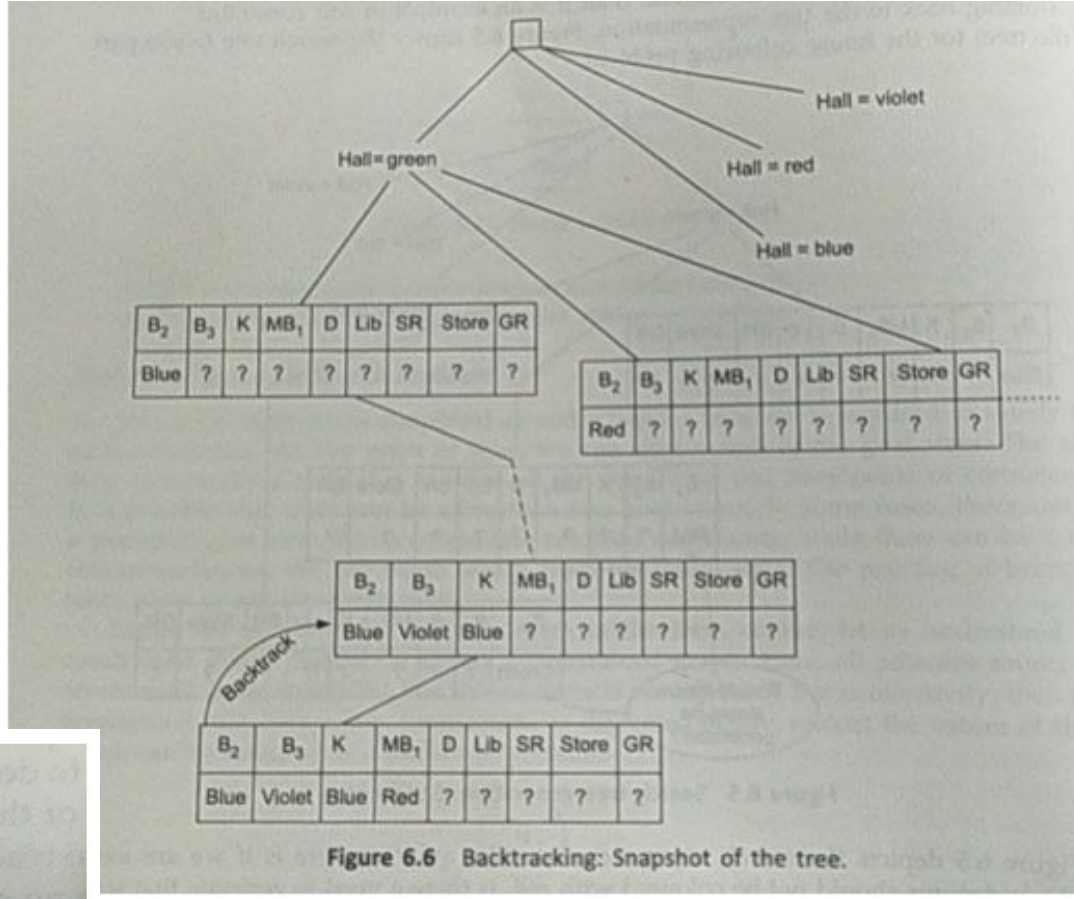


Figure 6.6 Backtracking: Snapshot of the tree.

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)

1	NC	1	1	
	S	E	N	D
	9	5	6	7
	M	O	R	E
	1	0	8	5
M	O	N	E	Y
1	0	6	5	2

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

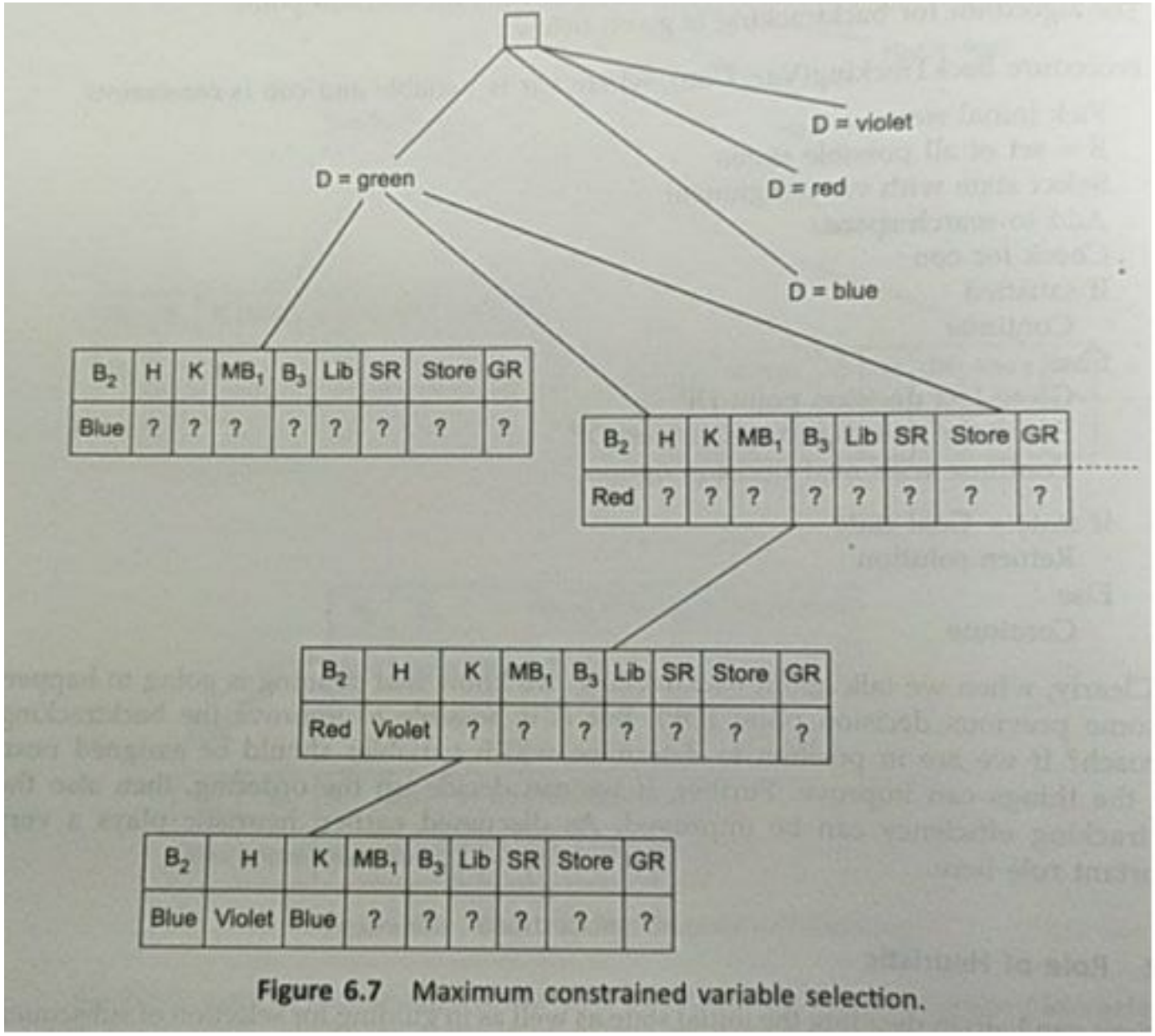


Figure 6.7 Maximum constrained variable selection.

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

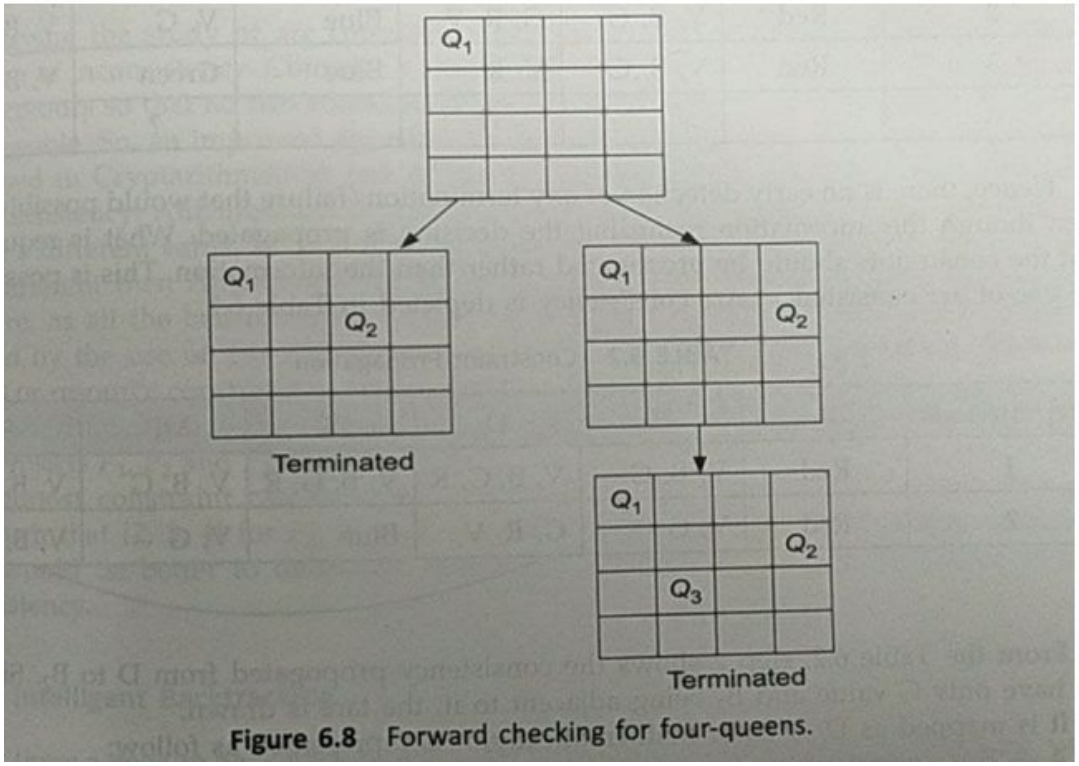


Figure 6.8 Forward checking for four-queens.

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
			?		?	