



ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

21CS54

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Course Code	21CS54	CIE Marks	50
Teaching Hours/Week (L:T:P: S)	3:0:0:0	SEE Marks	50
Total Hours of Pedagogy	40	Total Marks	100
Credits	03	Exam Hours	03

Course Learning Objectives

- CLO 1. Gain a historical perspective of AI and its foundations
- CLO 2. Become familiar with basic principles of AI toward problem solving
- CLO 3. Familiarize with the basics of Machine Learning & Machine Learning process, basics of Decision Tree, and probability learning
- CLO 4. Understand the working of Artificial Neural Networks and basic concepts of clustering algorithms

Module-1

Introduction: What is AI? Foundations and History of AI

Problem-solving: Problem-solving agents, Example problems, Searching for Solutions, Uninformed Search Strategies: Breadth First search, Depth First Search,

Textbook 1: Chapter 1- 1.1, 1.2, 1.3

Textbook 1: Chapter 3- 3.1, 3.2, 3.3, 3.4.1, 3.4.3

Teaching-Learning Process

Chalk and board, Active Learning. Problem based learning

Module-2

Informed Search Strategies: Greedy best-first search, A*search, Heuristic functions.
Introduction to Machine Learning , Understanding Data

Textbook 1: Chapter 3 - 3.5, 3.5.1, 3.5.2, 3.6

Textbook 2: Chapter 1 and 2

Manasa Sandeep, Dept. of CSE, DSATM

Teaching-Learning Process

Chalk and board, Active Learning, Demonstration

Module-3

Basics of Learning theory
Similarity Based Learning
Regression Analysis

Textbook 2: Chapter 3 - 3.1 to 3.4, Chapter 4, chapter 5.1 to 5.4

Teaching-Learning Process

Chalk and board, Problem based learning, Demonstration

Module-4

Decision Tree learning
Bayesian Learning

Textbook 2: Chapter 6 and 8

Teaching-Learning Process

Chalk and board, Problem based learning, Demonstration

Module-5

Artificial neural Network
Clustering Algorithms

Textbook 2: Chapter 10 and 13

Teaching-Learning Process

Chalk and board, Active Learning.

Textbooks

1. Stuart J. Russell and Peter Norvig, Artificial Intelligence, 3rd Edition, Pearson, 2015
2. S. Sridhar, M Vijayalakshmi "Machine Learning". Oxford, 2021

Reference:

1. Elaine Rich, Kevin Knight, Artificial Intelligence, 3rd edition, Tata McGraw Hill, 2013
2. George F Luger, Artificial Intelligence Structure and strategies for complex, Pearson Education, 5th Edition, 2011
3. Tom Michel, Machine Learning, McGrawHill Publication.

Weblinks and Video Lectures (e-Resources):

1. <https://www.kdnuggets.com/2019/11/10-free-must-read-books-ai.html>
2. <https://www.udacity.com/course/knowledge-based-ai-cognitive-systems--ud409>
3. <https://nptel.ac.in/courses/106/105/106105077/>
4. <https://www.javatpoint.com/history-of-artificial-intelligence>
5. <https://www.tutorialandexample.com/problem-solving-in-artificial-intelligence>
6. <https://techvidvan.com/tutorials/ai-heuristic-search/>
7. <https://www.analyticsvidhya.com/machine-learning/>
8. <https://www.javatpoint.com/decision-tree-induction>
9. <https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/ml-decision-tree/tutorial/>
10. <https://www.javatpoint.com/unsupervised-artificial-neural-networks>

Activity Based Learning (Suggested Activities in Class)/ Practical Based learning

Role play for strategies– DFS & BFS, Outlier detection in Banking and insurance transaction for identifying fraudulent behaviour etc. Uncertainty and reasoning Problem- reliability of sensor used to detect pedestrians using Bayes Rule

Assessment Details (both CIE and SEE)

The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the maximum marks (20 marks). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each subject/course if the student secures not less than 35% (18 Marks out of 50) in the semester-end examination (SEE), and a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation) and SEE (Semester End Examination) taken together

Continuous Internal Evaluation:

Three Unit Tests each of **20 Marks (duration 01 hour)**

1. First test at the end of 5th week of the semester
2. Second test at the end of the 10th week of the semester
3. Third test at the end of the 15th week of the semester

Two assignments each of **10 Marks**

4. First assignment at the end of 4th week of the semester
5. Second assignment at the end of 9th week of the semester

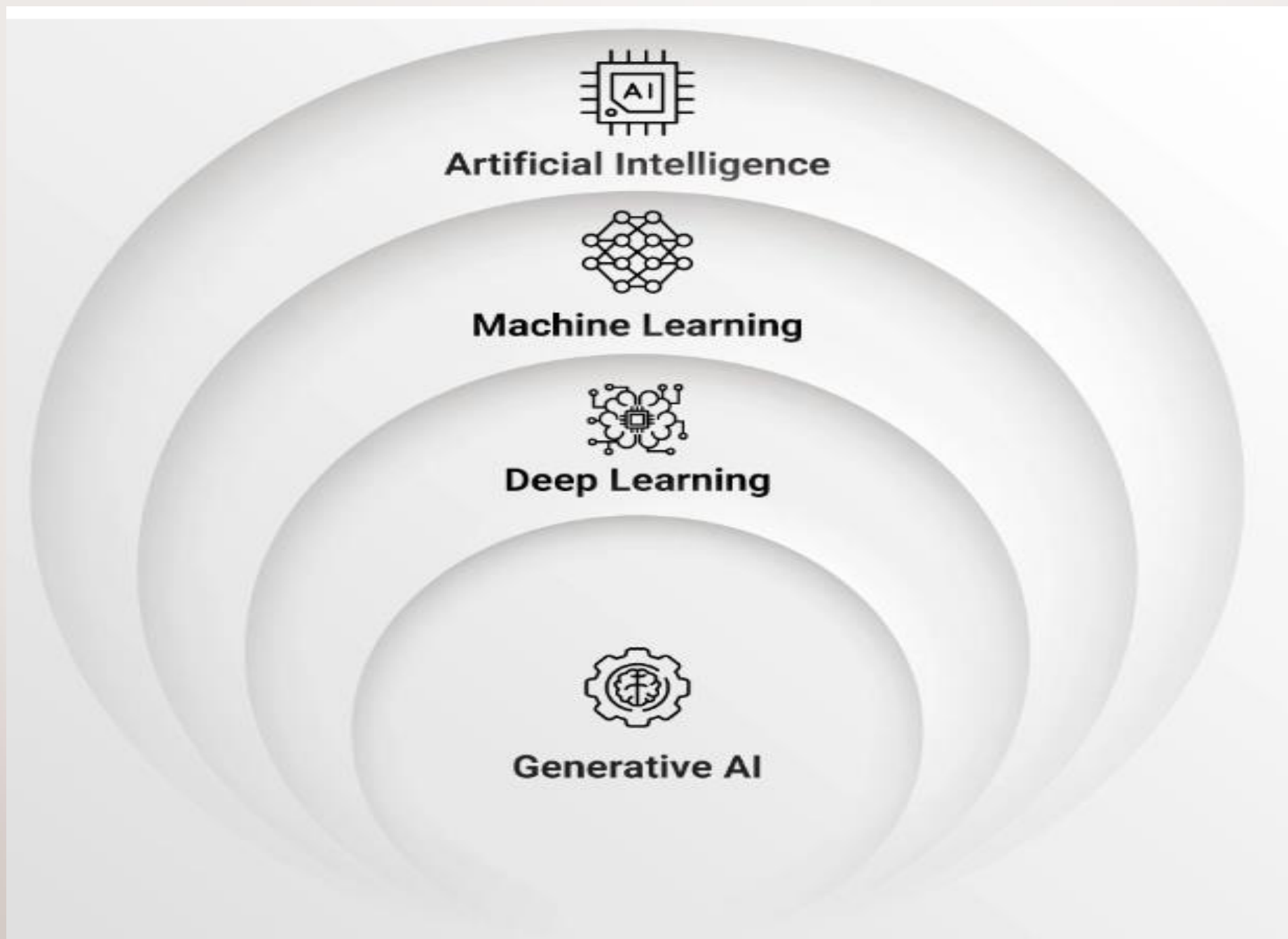
Group discussion/Seminar/quiz any one of three suitably planned to attain the COs and POs for **20 Marks (duration 01 hours) OR** Suitable Programming experiments based on the syllabus contents can be given to the students to submit the same as laboratory work(for example; Implementation of concept learning, implementation of decision tree learning algorithm for suitable data set, etc...)

6. At the end of the 13th week of the semester

The sum of three tests, two assignments, and quiz/seminar/group discussion will be out of 100 marks and will be **scaled down to 50 marks**

- AI
- ML
- DL
- G-AI

????????????????



MODULE 1

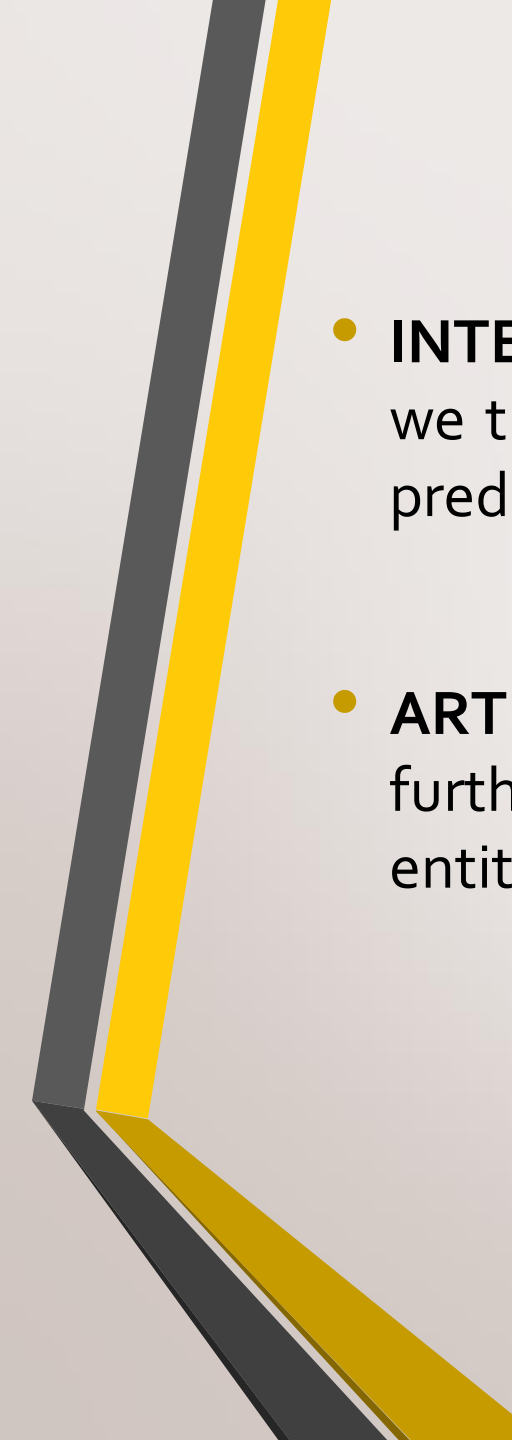
What is artificial intelligence?

- It is a branch of Computer Science that pursues creating the computers or machines as intelligent as human beings.

Example:
cocktail-party effect

« The capacity given by humans to machines to memorize and learn from experience, to think and create, to speak, to judge and make decisions »



- 
- **INTELLIGENCE:** For thousands of years, we have tried to understand how we think; that is, how a mere handful of matter can perceive, understand, predict, and manipulate a world far larger and more complicated than itself
 - **ARTIFICIAL INTELLIGENCE:** The field of artificial intelligence, or AI, goes further still. It attempts not just to understand but also to build intelligent entities.

What is AI?

Thinking Humanly “The exciting new effort to make computers think . . . machines with minds, in the full and literal sense.” (Haugeland, 1985) “[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . .” (Bellman, 1978)	Thinking Rationally “The study of mental faculties through the use of computational models.” (Charniak and McDermott, 1985) “The study of the computations that make it possible to perceive, reason, and act.” (Winston, 1992)
Acting Humanly “The art of creating machines that perform functions that require intelligence when performed by people.” (Kurzweil, 1990) “The study of how to make computers do things at which, at the moment, people are better.” (Rich and Knight, 1991)	Acting Rationally “Computational Intelligence is the study of the design of intelligent agents.” (Poole <i>et al.</i> , 1998) “AI . . . is concerned with intelligent behavior in artifacts.” (Nilsson, 1998)
Figure 1.1 Some definitions of artificial intelligence, organized into four categories.	

8 definitions in two dimensions:

Top: thought processes & reasoning

Bottom: behavior

Left: measures success in terms of fidelity to human performance

Right: measures rationality, A system is rational if it does the “right thing,” given what it knows

Acting humanly: The Turing Test approach

- Proposed by Alan Turing (1950)
- A computer passes the test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a person or from a computer
- To pass the test, The computer would need to possess the following capabilities:
 - **natural language processing** to enable it to communicate successfully in English;
 - **knowledge representation** to store what it knows or hears;
 - **automated reasoning** to use the stored information to answer questions and to draw new conclusions;
 - **machine learning** to adapt to new circumstances and to detect and extrapolate patterns.

- Total Turing Test: avoids direct physical interaction between the interrogator and the computer as physical simulation of a person is unnecessary for intelligence.
- Total Turing Test includes a video signal so that the interrogator can test the subject's perceptual abilities, as well as the opportunity for the interrogator to pass physical objects "through the hatch."

To pass the total Turing Test, the computer will need

- Computer vision to perceive objects, and
- Robotics to manipulate objects and move about

Thinking humanly: The cognitive modeling approach

- If we are going to say that a given program thinks like a human, we must have some way of **determining how humans think.**
- **Introspection**—trying to catch our own thoughts as they go by
- **Psychological experiments**—observing a person in action
- Once we have a sufficiently precise theory of the mind, it becomes possible to express the **theory as a computer program.** If the **program's input-output behavior matches corresponding human behavior,** that is evidence that some of the program's mechanisms could also be operating in humans
- **Allen Newell and Herbert Simon,** developed GPS, the “General Problem Solver” (1961), were not content merely to have their program solve problems correctly. **They were more concerned with comparing the trace of its reasoning steps to traces of human subjects solving the same problems.**

Thinking rationally: The “laws of thought” approach

- The Greek philosopher Aristotle was one of the first to attempt to codify “right thinking”: irrefutable **reasoning** processes
- His **syllogisms** provided patterns for argument structures that always yielded correct conclusions when given correct premises
- **“Socrates is a man; all men are mortal; therefore, Socrates is mortal.”**
- These laws of thought were supposed to govern the operation of the mind; their study initiated the field called **logic**.

Acting rationally: The rational agent approach

- An agent is just **something that acts**
- Computer agents are expected
 - operate autonomously
 - perceive their environment
 - persist over a prolonged time period
 - adapt to change
 - create and pursue goals.
- A **rational agent** is one that acts so as to achieve the **best outcome or, when there is uncertainty, the best expected outcome**

THE FOUNDATIONS OF ARTIFICIAL INTELLIGENCE

- ????

THE FOUNDATIONS OF ARTIFICIAL INTELLIGENCE

1. Philosophy
2. Mathematics
3. Economics
4. Neuroscience
5. Psychology
6. Computer engineering
7. Control theory and cybernetics
8. Linguistics

THE FOUNDATIONS OF ARTIFICIAL INTELLIGENCE

1. Philosophy

Philosophers (going back to 400 B.C.) made AI conceivable by **considering the ideas that the mind is in some ways like a machine,** that it operates on knowledge encoded in some internal language, and that thought can be used to choose what actions to take.

The following questions were answered:

- Can formal rules be used to draw valid conclusions?
- How does the mind arise from a physical brain?
- Where does knowledge come from?
- How does knowledge lead to action

- Aristotle (384–322 B.C.) formulated a precise set of laws governing the rational part of the mind. He developed an informal system of syllogisms for proper reasoning
- Concept of **rationalism** and **dualism** was proposed by Descartes (1596–1650)
- Rationalism is the knowledge that is derived from reason and logic
- Dualism: the mind (or the soul) is comprised of a non-physical substance, while the body is constituted of the physical substance known as **matter**
- Materialism: The theory or belief that nothing exists except matter and its movements and modifications.
- Empiricism: is the knowledge that is derived from experience and experimentation.
- Aristotle's algorithm was implemented **2300 years later by Newell and Simon in their GPS program**. We would now call it a regression planning system

2. Mathematics

Mathematicians **provided the tools to manipulate statements of logical certainty as well as uncertain, probabilistic statements.** They also set the groundwork for **understanding computation and reasoning about algorithms.**

The following are the questions addressed by mathematicians:

- What are the formal rules to draw valid conclusions?
- What can be computed?
- How do we reason with uncertain information?

Their work focused on 3 fundamental areas: **logic, computation, and probability.**

- George Boole (1815–1864) began the mathematical development of formal **logic proposed by philosophers of ancient Greece**
- The next step was to determine the **limits of what could be done with logic and computation.** The first algorithm is thought to be **Euclid's algorithm for computing greatest common divisors**

- Then came **incompleteness theorem:** showed that in any formal theory there are true statements that are undecidable in the sense that they have no proof within the theory.
- Motivated by this Alan Turing (1912–1954) tried to characterize exactly which functions are computable—**capable of being computed.**
- Turing also showed that there were some functions that no Turing machine can compute. For example, no machine can tell in general whether a given program will return an answer on a given input or run forever → **Halting Problem**
- Besides logic and computation, the **third great contribution of mathematics to AI is the theory of probability.** The Italian Gerolamo Cardano (1501–1576) first framed the **idea of probability, describing it in terms of the possible outcomes of gambling events.**

- James Bernoulli (1654–1705), Pierre Laplace (1749–1827), and others advanced the theory and introduced new statistical methods.
- Thomas Bayes (1702–1761), proposed a rule for updating probabilities in the light of new evidence. Bayes' rule underlies most modern approaches to uncertain reasoning in AI systems

3. Economics

Economists **formalized the problem of making decisions that maximize the expected outcome to the decision maker**

The following questions were addressed

- How should we make decisions so as to maximize payoff?
- How should we do this when others may not go along?
- How should we do this when the payoff may be far in the future?
- Scottish **philosopher Adam Smith** was the first to treat it as a science, economies can be thought of as consisting of individual agents maximizing their own economic well-being.
- The mathematical treatment of “**preferred outcomes**” or utility was first formalized by L’eon Walras (1834-1910) and was improved by **John von Neumann** and Oskar Morgenstern in their book **The Theory of Games and Economic Behavior (1944).**

Von Neumann and Morgenstern's development of **game theory** included the surprising result that, for some games, a rational agent should adopt policies that are randomized.

The work of **Richard Bellman** (1957) formalized a class of sequential decision problems called **Markov decision processes**

Work in **economics** and **operations research** has contributed much to our notion of rational agents

The pioneering **AI researcher Herbert Simon** (1916–2001) won the Nobel Prize in economics in 1978 for his early work showing that models based on **satisficing**—making decisions that are “**good enough**,” rather than **laboriously calculating an optimal decision**—gave a better description of actual human behavior

4. Neuro Science

Neuroscientists discovered some facts about how the brain works and the ways in which it is similar to and different from computers

	Supercomputer	Personal Computer	Human Brain
Computational units	10^4 CPUs, 10^{12} transistors	4 CPUs, 10^9 transistors	10^{11} neurons
Storage units	10^{14} bits RAM 10^{15} bits disk	10^{11} bits RAM 10^{13} bits disk	10^{11} neurons 10^{14} synapses
Cycle time	10^{-9} sec	10^{-9} sec	10^{-3} sec
Operations/sec	10^{15}	10^{10}	10^{17}
Memory updates/sec	10^{14}	10^{10}	10^{14}

How do brains process information?

Aristotle wrote, “Of all the animals, man has the largest brain in proportion to his size.”. Still, it was not until the middle of the 18th century that the brain was widely recognized as the seat of consciousness

- Computers have a cycle time that is a million times faster than a brain. The brain makes up for that with far more storage and interconnection than even a high-end personal computer
- Even with a computer of virtually unlimited capacity, we still would not know how to achieve the brain's level of intelligence.

5. Psychology

- How do humans and animals think and act?

Psychologists adopted the idea that humans and animals can be considered information processing machines

Cognitive science is the study of behavior of humans and animals

Theory said that **“a cognitive theory should be like a computer program”** (Anderson, 1980); that is, it should describe a detailed information processing mechanism whereby some cognitive function might be implemented

Computer engineers provided the ever-more-powerful machines that make AI applications possible.

6. Computer Engineering

- How can we build an efficient computer?
- For artificial intelligence to succeed, we need two things: **intelligence and an artifact**. The **computer has been the artifact** of choice.
- The **modern digital electronic computer** was invented independently and almost simultaneously by **scientists in three countries** embattled in World War II.
- Babbage's **Analytical Engine** included addressable memory, stored programs, and conditional jumps and was the first **artifact capable of universal computation**. Babbage's colleague **Ada Lovelace**, was perhaps the **world's first programmer**

7. Control theory and cybernetics

- How can artifacts operate under their own control?

Control theory deals with **designing devices that act optimally on the basis of feedback from the environment.** Initially, the mathematical tools of control theory were quite different from AI, but the fields are coming closer together.

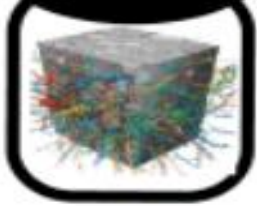
8. Linguistics

- How does language relate to thought?

Linguists showed that language use fits into this model

- The problem of understanding language soon turned out to be considerably **more complex.**
- Understanding language requires an **understanding of the subject matter and context, not just an understanding of the structure of sentences.**

1943



Evolution of
Artificial
neurons

1950



Turing
Machine

1956



Birth of AI:
Dartmouth
Conference

1966



First Chatboat :
ELIZA

1972



First
Intelligence
Robot :
WABOT -1

1974-1980



First AI
winer

1980



Expert
Ststem

1987-1993



Second AI
Winer

1997



IBM Deep blue
: first computer
to beat a world
chess champion

2002



AI in Home:
Roomba

2011



IBM s Watson :
Wins a quiz
show

2012



Google now

2014



Chatbot Eugene
Goostman:Wines
a "Turing test

2015



Amazon
Echo

THE HISTORY OF ARTIFICIAL INTELLIGENCE

- The gestation of artificial intelligence (1943–1955)
- The birth of artificial intelligence (1956)
- Early enthusiasm, great expectations (1952–1969)
- A dose of reality (1966–1973)
- Knowledge-based systems: The key to power? (1969–1979)
- AI becomes an industry (1980–present)
- The return of neural networks (1986–present)
- AI becomes a science
- The emergence of intelligent agents (1995–present)

THE HISTORY OF ARTIFICIAL INTELLIGENCE

- The history of AI has had cycles of
 - Success
 - Misplaced optimism, and resulting cutbacks in enthusiasm and funding.

The gestation of artificial intelligence (1943–1955)

- First work → First AI neuron was developed by Warren McCulloch and Walter Pitts (1943)
- Using 3 sources: knowledge of the basic physiology and function of neurons in the brain; a formal analysis of propositional logic due to Russell and Whitehead; and Turing's theory of computation
- They proposed a model of artificial neurons in which each neuron is characterized as being "on" or "off," with a switch to "on" occurring in response to stimulation by a sufficient number of neighboring neurons

- All logical connectives (and, or, not, etc.) could be implemented by simple net structures
- Donald Hebb (1949) demonstrated a simple updating rule for modifying the connection strengths between neurons. His rule, now called Hebbian learning
- There were a number of early examples of work that can be characterized as AI, but Alan Turing's vision was perhaps the most influential. He gave lectures on the topic as early as 1947 at the London Mathematical Society and articulated a persuasive agenda in his 1950 article "Computing Machinery and Intelligence."
- Therein, he introduced the Turing Test, machine learning, genetic algorithms, and reinforcement learning.

The birth of artificial intelligence (1956)

- John McCarthy along with Minsky, Claude Shannon, and Nathaniel Rochester organized a **two-month workshop at Dartmouth in the summer of 1956**.
- Allen Newell and Herbert Simon invented a computer program capable of thinking non-numerically, and thereby solved the venerable mind–body problem
- John McCarthy coined the term “Artificial Intelligence” in this event
- The gathered researchers and scientists explored the possibility of creating machines that could simulate human intelligence and concluded that **AI has to become a separate field of study**
- The conference resulted in increased research, funding, and interest in the pursuit of creating intelligent machines.

Early enthusiasm, great expectations (1952–1969)

The period was marked by optimism and ambitious goals. During this time frame, researchers and scientists were fueled by the belief that they could create machines capable of intelligent behavior and problem-solving.

Some key highlights from this period:

- **Logic Theorist (1956):** In 1956, Allen Newell and Herbert A. Simon developed the Logic Theorist, considered one of the first artificial intelligence programs. It could prove mathematical theorems using symbolic logic.
- **General Problem Solver (1957):** Newell, Simon, and J.C. Shaw came up with General Problem Solver (GPS), a more generalized version of the Logic Theorist. It aimed to solve a wide range of problems, applying heuristics and searching through possible solutions. Then they formulated the famous **physical symbol system hypothesis**

Early enthusiasm, great expectations (1952–1969)

- **Physical symbol system hypothesis**, states that “**a physical symbol system has the necessary and sufficient means for general intelligent action.**” What they meant is that any system (human or machine) **exhibiting intelligence must operate by manipulating data structures composed of symbols.**
- **Language Translation and Eliza (1966):** The Georgetown-IBM experiment in 1954 demonstrated early efforts in machine translation. Additionally, in 1966, Joseph Weizenbaum created **ELIZA**, an early natural language processing program that simulated conversation.

Challenges and Setbacks: Despite the early successes, the field faced challenges. The high expectations set by the pioneers **often led to disappointment**, and progress was slower than initially anticipated. **Funding and interest in AI research declined towards the end of the 1960s.**

A dose of reality (1966–1973)

- This period was characterized by a more sober assessment of the field's progress and capabilities. During this time frame, researchers faced challenges and setbacks that prompted a **reevaluation of the initial optimism**.
- **Beginning of AI Winter:** The term "AI winter" refers to a **series of periods where interest and funding in artificial intelligence research significantly declined**. The first AI winter started around the **late 1960s to early 1970s**. This was partly due to unmet expectations, **as early AI systems struggled with limitations in computational power, memory, and the ability to handle real-world complexity**.

Knowledge-based systems: The key to power? (1969–1979)

- This period reflects a shift in focus **towards knowledge-based systems**. They aimed to capture and utilize expert knowledge to solve specific problems.
- **Expert Systems Development:** These systems were designed to emulate the decision-making abilities of human experts in specific domains. Researchers believed that by encoding expert knowledge into a computer, they could address complex problems in fields such as medicine, finance, and engineering.
- **MYCIN (1972):** was an expert systems developed for diagnosing bacterial infections, Created by Edward Shortliffe. MYCIN demonstrated the potential of knowledge-based systems in the medical domain.
- **DENDRAL (1965–1982):** DENDRAL, developed by Edward Feigenbaum and Joshua Lederberg, was another pioneering expert system. It focused on analyzing mass spectrometry data for organic chemical compound identification, showcasing the applicability of knowledge-based approaches to scientific domains.

The Birth of AI as an Industry:(1980–present)

- The development of knowledge-based systems contributed to the recognition of **AI as an industry**.
- AI transitioned from primarily academic research to a flourishing industry with practical applications and commercialization.
- **Expert Systems and Commercialization:** The 1980s saw the commercialization of expert systems and the application of AI technologies in various industries. Companies began to invest in AI for practical problem-solving and decision support in areas such as finance, healthcare, and manufacturing.
- **AI Winter (Late 1980s to Early 1990s):** The initial enthusiasm in the early 1980s was again followed by "AI winter," where progress in **AI research slowed, and funding declined. Unrealistic expectations, coupled with technical challenges and over-promising, contributed to this downturn.**

The return of neural networks (1986–present)

- It refers to the Significant resurgence and sustained interest in neural network research and development
- In the 1990s and beyond, development of neural networks made researchers to shift the focus to data-driven approaches leading to many breakthroughs in areas such as natural language processing and computer vision marking the rise of machine learning

AI becomes a science

The phase generally refers to the maturation of artificial intelligence (AI) as an **academic discipline and a scientific field of study**. This evolution involves the establishment of foundational principles, theoretical frameworks, and systematic approaches to understanding and advancing AI.

The emergence of intelligent agents (1995–present)

- Intelligent agents are autonomous entities that perceive their environment, make decisions, and take actions to achieve goals.
- The period from 1995 to the present has seen a rapid evolution in the capabilities and applications of intelligent agents due to the continued development and deployment of intelligent agents in the field of artificial intelligence (AI)

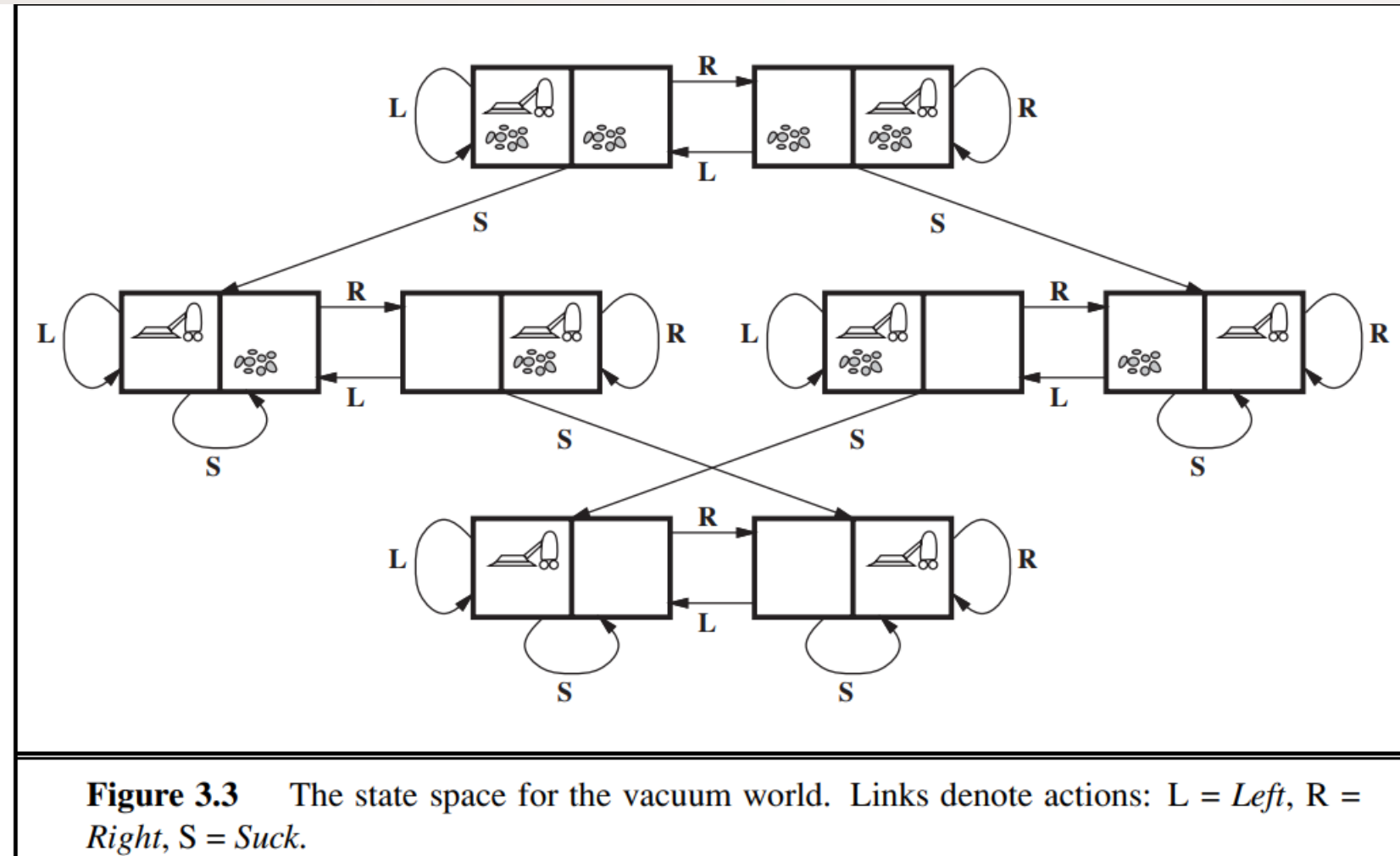
EXAMPLE PROBLEMS

- Problems can be
 1. Toy Problem: (vacuum world, 8-puzzle, 8-queens problem)
 2. A Real World Problem: (route-finding problem, touring problem, TSP, VLSI layout problem, Robot navigation, Automatic assembly sequencing, protein design)
- A TOY PROBLEM is intended to illustrate or exercise various problem-solving methods. It can be given a **concise, exact description** and hence is **usable by different researchers to compare the performance of algorithms**.
- 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 problems

- **Vacuum World Problem**

- **States:** The state is determined by both the agent location and the dirt locations. The 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 state can be designated as the initial state.
- **Actions:** In this simple environment, each state has just three actions: *Left*, *Right*, and *Suck*. Larger environments might also include *Up* and *Down*.
- **Transition model:** The actions have their expected effects, except that moving *Left* in the leftmost square, moving *Right* in the rightmost square, and *Sucking* in a clean square have no effect. The complete state space is shown in Figure 3.3.
- **Goal test:** This checks whether all the squares are clean.
- **Path cost:** Each step costs 1, so the path cost is the number of steps in the path.



this toy problem has discrete locations, discrete dirt, reliable cleaning, and it never gets any dirtier.

8-puzzle Problem

- It 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. The standard formulation is as follows:

- **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. Note that any given goal can be reached from exactly half of the possible initial states (Exercise 3.4).
- **Actions:** The simplest formulation defines the actions as movements of the blank space *Left*, *Right*, *Up*, or *Down*. Different subsets of these are possible depending on where the blank is.
- **Transition model:** Given a state and action, this returns the resulting state; for example, if we apply *Left* to the start state in Figure 3.4, the resulting state has the 5 and the blank switched.
- **Goal test:** This checks whether the state matches the goal configuration shown in Figure 3.4. (Other goal configurations are possible.)
- **Path cost:** Each step costs 1, so the path cost is the number of steps in the path.

8-puzzle Problem

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

A typical instance of the 8-puzzle.

8-queens problem

- The goal of the 8-queens problem is to place eight queens on a chessboard such that no queen attacks any other

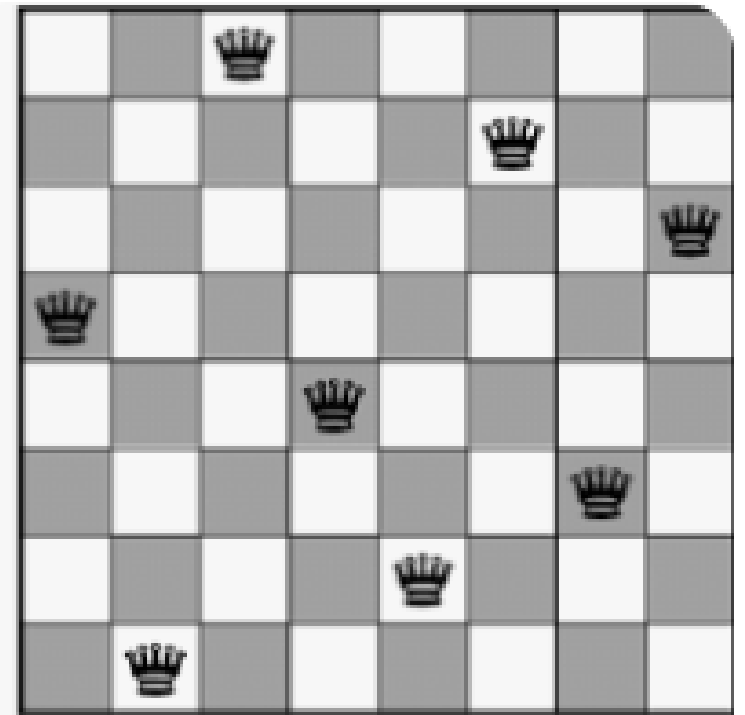
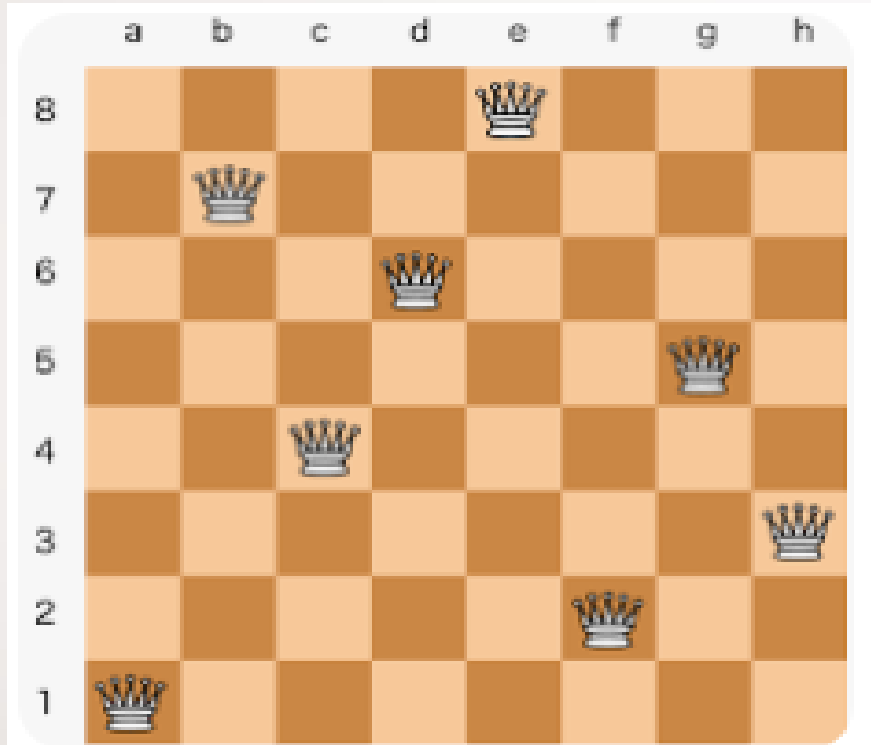
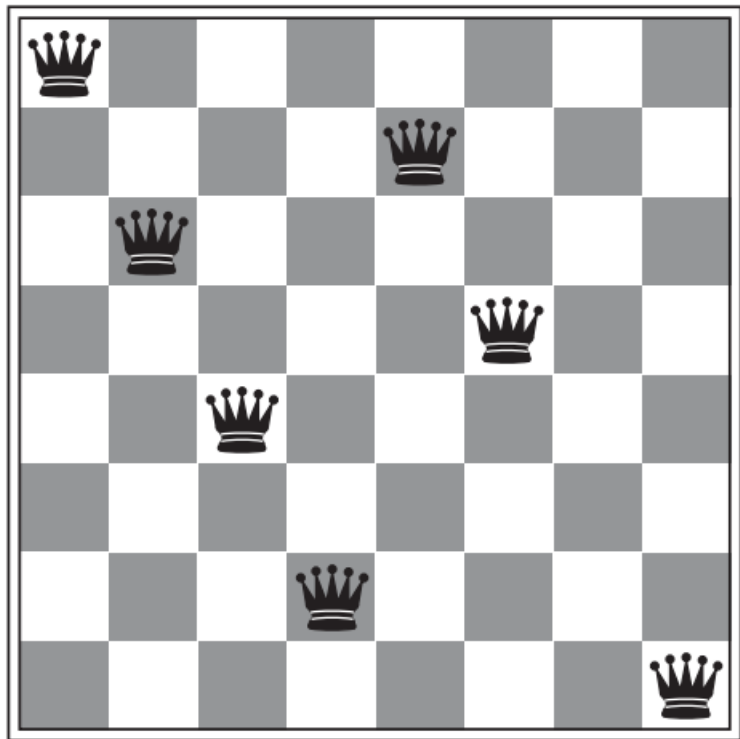
Two types of formulation :

- An INCREMENTAL FORMULATION involves operators that augment the state description, starting with an **empty state**; i.e, **each action adds a queen to the state**.
- A **complete-state formulation** starts with all 8 queens on the board and moves them around. In either case, the path cost is of no interest because only the final state counts.

In this formulation, we have $64 \cdot 63 \cdots 57 \approx 1.8 \times 10^{14}$ possible sequences to investigate. A better formulation would prohibit placing a queen in any square that is already attacked:

- **States:** All possible arrangements of n queens ($0 \leq n \leq 8$), one per column in the leftmost n columns, with no queen attacking another.
- **Actions:** Add a queen to any square in the leftmost empty column such that it is not attacked by any other queen.

This formulation reduces the 8-queens state space from 1.8×10^{14} to just 2,057, and solutions are easy to find. On the other hand, for 100 queens the reduction is from roughly 10^{400} states to about 10^{52} states (Exercise 3.5)—a big improvement, but not enough to make the problem tractable. Section 4.1 describes the complete-state formulation, and Chapter 6 gives a simple



Real-world problems

- Route-finding problem
- Applications: routing video streams in computer networks, military operations planning, and airline travel-planning systems, involve much more complex specifications
- Consider the airline travel problems that must be solved by a travel-planning

Real-world problems

- **States:** Each state obviously includes a location (e.g., an airport) and the current time. Furthermore, because the cost of an action (a flight segment) may depend on previous segments, their fare bases, and their status as domestic or international, the state must record extra information about these “historical” aspects.
- **Initial state:** This is specified by the user’s query.
- **Actions:** Take any flight from the current location, in any seat class, leaving after the current time, leaving enough time for within-airport transfer if needed.
- **Transition model:** The state resulting from taking a flight will have the flight’s destination as the current location and the flight’s arrival time as the current time.
- **Goal test:** Are we at the final destination specified by the user?
- **Path cost:** This depends on monetary cost, waiting time, flight time, customs and immigration procedures, seat quality, time of day, type of airplane, frequent-flyer mileage awards, and so on.

Real-world problems

Touring problem:

- Visit every city given at least once, starting and ending in same city.

Traveling salesperson problem (TSP):

- Is a touring problem in which each city must be visited exactly once. The aim is to find the **shortest tour**

A VLSI layout problem

- requires positioning **millions of components and connections** on a chip to minimize area, minimize circuit delays, minimize stray capacitances, and maximize manufacturing yield.

The layout problem comes after the logical design phase and is usually split into two parts:

cell layout and channel routing

- In cell layout, the primitive components of the circuit **are grouped into cells**, each of which performs some recognized function. Each cell has a fixed footprint (size and shape) and requires a certain number of connections to each of the other cells. The aim is to place the cells on the chip so **that they do not overlap** and so that there is **room for the connecting wires to be placed between the cells**.
- Channel routing finds a specific route for each wire through the gaps between the cells.

Robot navigation

- Is a generalization of the route-finding problem
- Robot moves in a continuous space with (in principle) an infinite set of possible actions and states.
- For a circular robot moving on a **flat surface**, the space is essentially **two-dimensional**.
- When the robot **has arms and legs or wheels that must** also be controlled, the search space becomes **many-dimensional**.
- Advanced techniques are required just to make the search space finite.

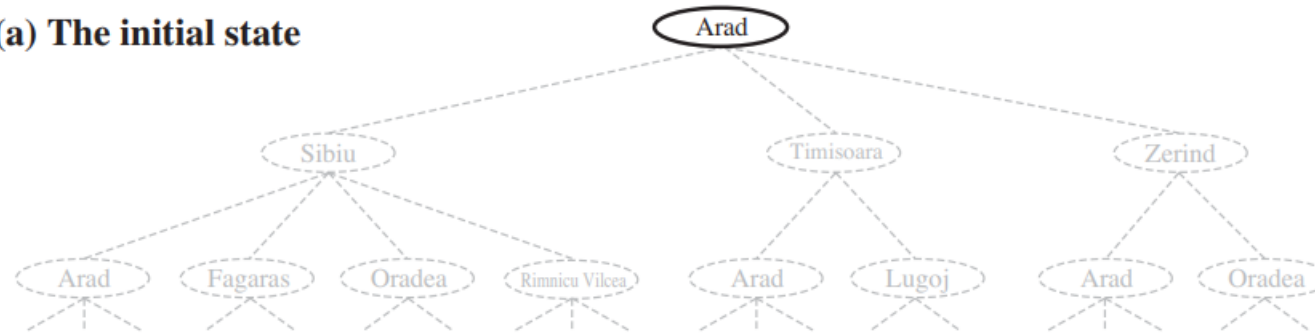
Automatic assembly sequencing

- of complex objects by a robot was first demonstrated by FREDDY (Michie, 1972).
- Progress since then has been slow but sure, to the point where the assembly of intricate objects such as electric motors is economically feasible.
- In assembly problems, the **aim is to find an order in which to assemble the parts of some object**. If the wrong order is chosen, there will be no way to **add some part later in the sequence without** undoing some of the work already done
- Another important assembly problem is **PROTEIN DESIGN**, in which the goal is to find a **sequence of amino acids** that will fold into a **three-dimensional protein** with the right properties to cure **some disease**

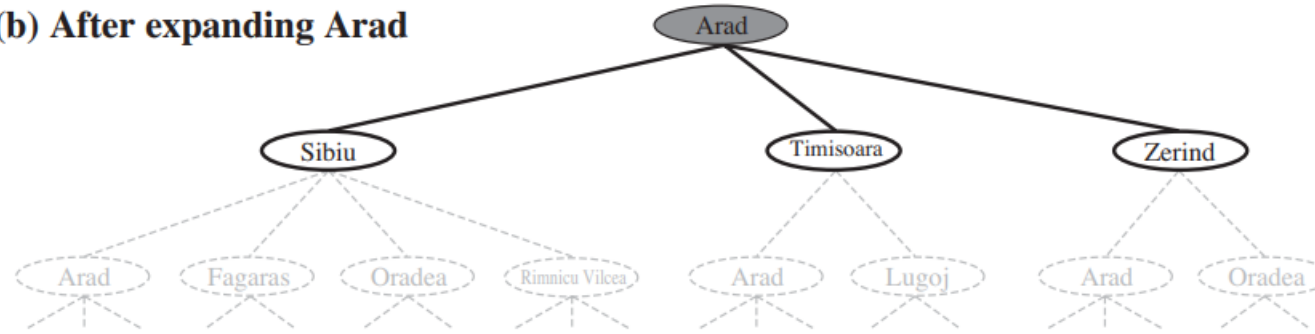
SEARCHING FOR SOLUTIONS

- After formulating the problems clearly and precisely, next step is to solve them i.e,
- Finding solution to the problems
- A solution is an **action sequence**, so search algorithms work by considering various possible action sequences
- The possible **action sequences** starting at the initial state form a search tree with the initial state at the root; the branches are actions and the nodes correspond to states in the state space of the problem.
- Then we need to consider taking various actions. That is **expanding** the current state; applying each legal action to the current state, thereby **generating** a new set of states.
- we add branches from the **parent** node leading to new child nodes
- A leaf node is a node with **no children** in the tree.
- Collection of nodes that have been generated but not yet expanded are called **fringe**

(a) The initial state



(b) After expanding Arad



(c) After expanding Sibiu

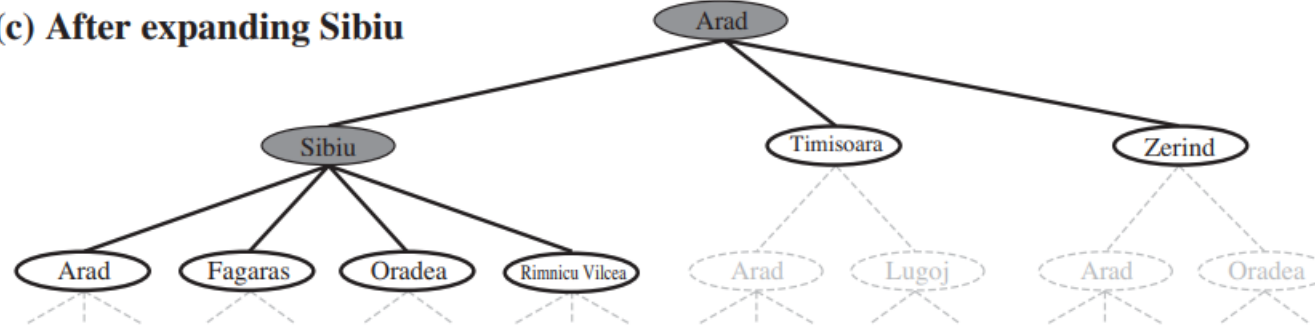


Figure 3.6 Partial search trees for finding a route from Arad to Bucharest. Nodes that have been expanded are shaded; nodes that have been generated but not yet expanded are outlined in bold; nodes that have not yet been generated are shown in faint dashed lines.

SEARCHING FOR SOLUTIONS

- The choice of which state to be expanded is determined by the **SEARCH STRATEGY**.
- Search algorithms require a data structure to keep track of the search tree that is being constructed. The node data structure contains the following 5 components

STATE: the state in the state space to which the node corresponds;

PARENT: the node in the search tree that generated this node;

ACTION: the action that was applied to the parent to generate the node;

PATH-COST: the cost, traditionally denoted by $g(n)$, of the path from the initial state to the node, as indicated by the parent pointers

DEPTH: the number of steps along the path from the initial state

SEARCHING FOR SOLUTIONS

function TREE-SEARCH(*problem*, *strategy*) **returns** a solution, or failure
 initialize the search tree using the initial state of *problem*
 loop do
 if there are no candidates for expansion **then return** failure
 choose a leaf node for expansion according to *strategy*
 if the node contains a goal state **then return** the corresponding solution
 else expand the node and add the resulting nodes to the search tree

Figure 3.7 An informal description of the general tree-search algorithm.

function TREE-SEARCH(*problem*, *fringe*) **returns** a solution, or failure

fringe \leftarrow INSERT(MAKE-NODE(INITIAL-STATE[*problem*]), *fringe*)

loop do

if EMPTY?(*fringe*) **then return** failure

node \leftarrow REMOVE-FIRST(*fringe*)

if GOAL-TEST[*problem*] applied to STATE[*node*] **succeeds**

then return SOLUTION(*node*)

fringe \leftarrow INSERT-ALL(EXPAND(*node*, *problem*), *fringe*)

function EXPAND(*node*, *problem*) **returns** a set of nodes

successors \leftarrow the empty set

for each \langle action, result \rangle **in** SUCCESSOR-FN[*problem*](STATE[*node*]) **do**

s \leftarrow a new NODE

STATE[*s*] \leftarrow result

PARENT-NODE[*s*] \leftarrow *node*

ACTION[*s*] \leftarrow action

PATH-COST[*s*] \leftarrow PATH-COST[*node*] + STEP-COST(*node*, action, *s*)

DEPTH[*s*] \leftarrow DEPTH[*node*] + 1

add *s* to *successors*

return *successors*

Figure 3.9 The general tree-search algorithm. (Note that the *fringe* argument must be an empty queue, and the type of the queue will affect the order of the search.) The SOLUTION function returns the sequence of actions obtained by following parent pointers back to the root.

Measuring problem-solving performance

- The algorithm's performance can be evaluated by following four ways:
 - **Completeness:** Is the algorithm guaranteed to find a solution when there is one?
 - **Optimality:** Does the strategy find the optimal solution, as defined on page 68?
 - **Time complexity:** How long does it take to find a solution?
 - **Space complexity:** How much memory is needed to perform the search?
- In AI, the graph is often represented by the initial state, actions, and transition model and is frequently infinite. For these reasons, complexity is expressed in terms of three quantities:
 - b , **branching factor** or maximum number of successors of any node;
 - d , **depth** of the **shallowest goal** node (i.e., the number of steps along the path from the root); and
 - m , the maximum length of any path in the state space.
- Time is often measured in terms of the number of nodes generated during the search, and space in terms of the maximum number of nodes stored in memory.

Uninformed Search Strategies

- Uninformed search is also called as blind search.
- These strategies have no additional information about states beyond that provided in the problem definition.
- All they can do is generate successors and distinguish a goal state from a non-goal state.
- All search strategies are distinguished by the order in which nodes are expanded.

Example:

BFS: Breadth-first search

DFS: Depth-first search

Breadth-first search

- Breadth-first search is a simple strategy in which the root node is expanded first, then all the successors of the root node are expanded next, then their successors and so on
- All the nodes are expanded at a given depth in the search tree before any nodes at the next level are expanded.
- It is an uninformed Search Technique
- Breadth-first search is an instance of the general graph-search algorithm in which the **shallowest** unexpanded node is chosen for expansion.
- Uses FIFO (Queue) data structure

Breadth-first search

- Uses FIFO (Queue) data structure
- BFS is **complete**: —if the shallowest goal node is at some finite depth d , breadth-first search will **eventually find it** after generating all shallower nodes
- BFS is **optimal**
- Time Complexity: An exponential complexity bound such as $O(b^d)$
- The memory requirements are a bigger problem for breadth-first search than is the execution time. Though time is still a major factor
- Exponential-complexity search problems cannot be solved by uninformed methods for any but the smallest instances

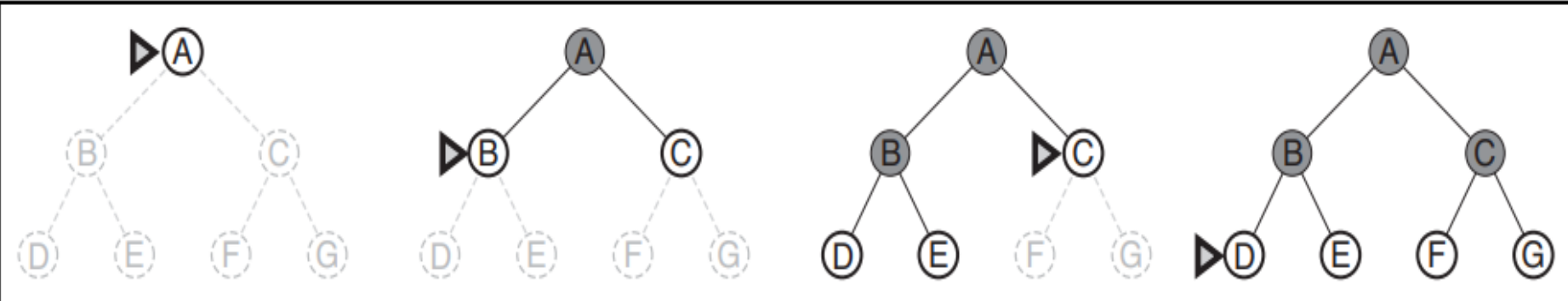


Figure 3.12 Breadth-first search on a simple binary tree. At each stage, the node to be expanded next is indicated by a marker.

Depth	Nodes	Time	Memory
2	110	.11 milliseconds	107 kilobytes
4	11,110	11 milliseconds	10.6 megabytes
6	10^6	1.1 seconds	1 gigabyte
8	10^8	2 minutes	103 gigabytes
10	10^{10}	3 hours	10 terabytes
12	10^{12}	13 days	1 petabyte
14	10^{14}	3.5 years	99 petabytes
16	10^{16}	350 years	10 exabytes

Figure 3.13 Time and memory requirements for breadth-first search. The numbers shown assume branching factor $b = 10$; 1 million nodes/second; 1000 bytes/node.

Depth-first search

- Depth-first search always expands the deepest node in the current fringe of the search tree
- The search proceeds immediately to the deepest level of the search tree, where the nodes have no successors.
- As those nodes are expanded, they are dropped from the fringe, so then the search “backs up” to the next deepest node that still has unexplored successors
- depth-first search uses a LIFO queue
- DFS has a very modest memory requirement
- It expands the **Deepest node**
- Hence, the algorithm is **Incomplete**, i.e. no guarantee of solution

Depth-first search

- Even when it finds the solution, **no guarantee** that it is **optimal solution**
- For a graph search, there is no advantage, but a depth-first tree search needs to **store only a single path from the root to a leaf node**, along with the remaining unexpanded sibling nodes for each node on the path. Once a node has been expanded, it can be removed from memory as soon as all its descendants have been fully explored.
- Drawback: It can make a wrong choice and get stuck going down a very long path when a different choice would lead to a solution near the root of the search tree

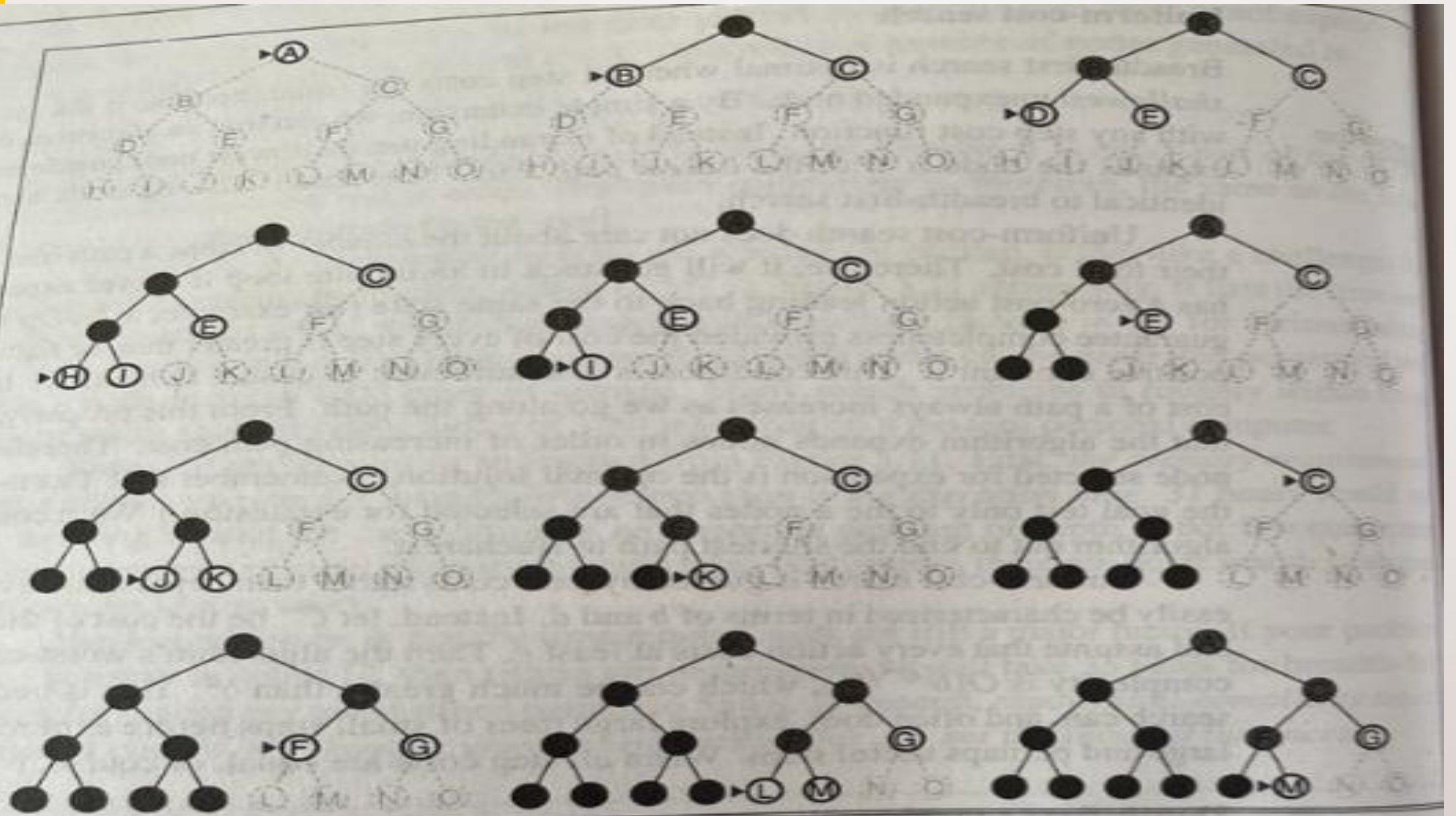


Figure 3.12 Depth-first search on a binary tree. Nodes that have been expanded and have no descendants in the fringe can be removed from memory; these are shown in black. Nodes at depth 3 are assumed to have no successors and *M* is the only goal node.

Activity- Role Play

- 1. Water Jug Problem
- 2. Missionaries and cannibals problem