

CST401 ARTIFICIAL INTELLIGENCE

MODULE 1



















Course Outcome

Course Outcomes: After the completion of the course the student will be able to

CO#	CO
CO1	Explain the fundamental concepts of intelligent systems and their architecture. (Cognitive Knowledge Level: Understanding)
CO2	Illustrate uninformed and informed search techniques for problem solving in intelligent systems. (Cognitive Knowledge Level: Understanding)
CO3	Solve Constraint Satisfaction Problems using search techniques. (Cognitive Knowledge Level: Apply)
CO4	Represent AI domain knowledge using logic systems and use inference techniques for reasoning in intelligent systems. (Cognitive Knowledge Level: Apply)
CO5	Illustrate different types of learning techniques used in intelligent systems (Cognitive Knowledge Level: Understand)

Mapping of course outcomes with program outcomes

Mapping of course outcomes with program outcomes

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1												
CO2												
CO3												
CO4												
CO5												

Abstract POs defined by National Board of Accreditation

Abstract POs defined by National Board of Accreditation			
PO#	Broad PO	PO#	Broad PO
PO1	Engineering Knowledge	PO7	Environment and Sustainability
PO2	Problem Analysis	PO8	Ethics
PO3	Design/Development of solutions	PO9	Individual and team work
PO4	Conduct investigations of complex problems	PO10	Communication
PO5	Modern tool usage	PO11	Project Management and Finance
PO6	The Engineer and Society	PO12	Life long learning

Assessment Pattern

Bloom's Category	Continuous Assessment Tests		End Semester Examination Marks (%)
	Test 1 (%)	Test 2 (%)	
Remember	30	30	30
Understand	60	30	40
Apply	20	40	30
Analyze			
Evaluate			
Create			

Mark Distribution

Mark Distribution

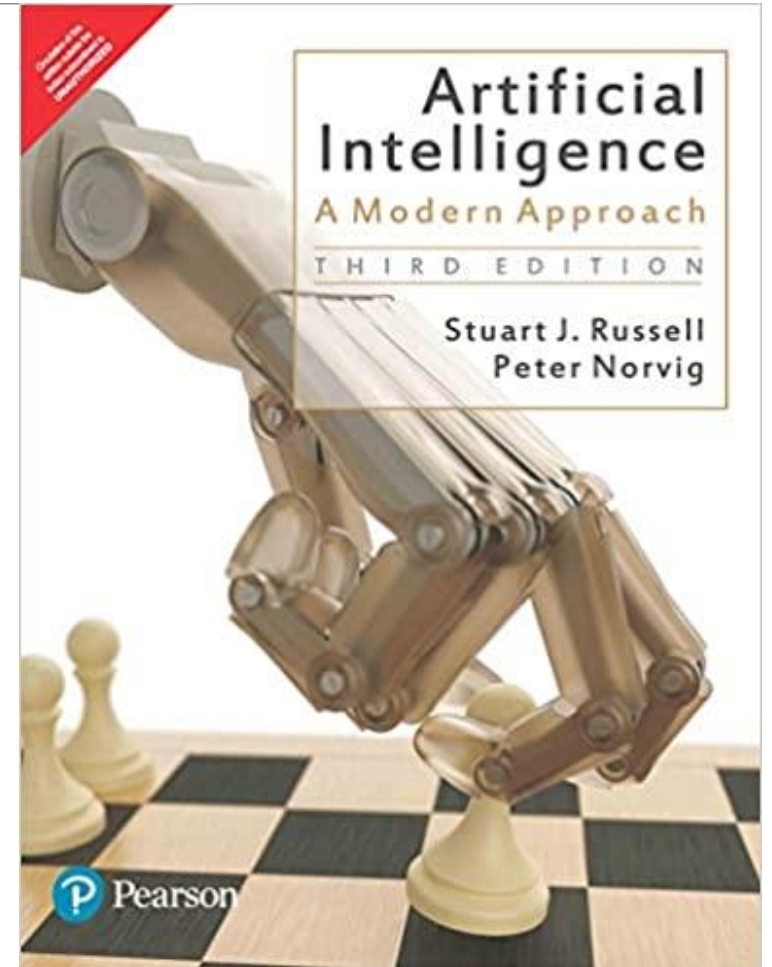
Total Marks	CIE Marks	ESE Marks	ESE Duration
150	50	100	3

Continuous Internal Evaluation Pattern:

Attendance	10 marks
Continuous Assessment Tests(Average of Series Tests 1 & 2)	25 marks
Continuous Assessment Assignment	15 marks

Textbook

Stuart Russell and Peter Norvig. **Artificial Intelligence: A Modern Approach**, 3rd Edition. Prentice Hall.



SYLLABUS

Introduction- What is Artificial Intelligence

Foundation of AI

History of AI

Applications of AI

Intelligent Agents- Agents and Environment

Good Behaviour

The concept of rationality, nature of environment

Structure of Agents

CO1: Explain the fundamental concepts of intelligent systems and their architecture.
(Cognitive Knowledge Level: Understanding)

PO1-Engineering Knowledge

ARTIFICIAL INTELLIGENCE

Artificial intelligence allows machines to replicate the capabilities of the human mind. From the development of self-driving cars to the development of smart assistants like Siri and Alexa, AI is a growing part of everyday life.

Artificial intelligence is a wide-ranging branch of computer science concerned with building smart machines capable of performing tasks that typically require human intelligence.

What is AI

Thinking Humanly “The exciting new effort to make computers think . . . <i>machines with minds</i> , 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.

The definitions on the left measure success in terms of **fidelity to human performance**,

the ones on the right measure against an ideal performance measure, called **rationality**.

A system is rational if it does the “right thing,” given what it knows.

Turing Test

(Human) judge communicates with a human and a machine over text-only channel,

Both human and machine try to act like a human,

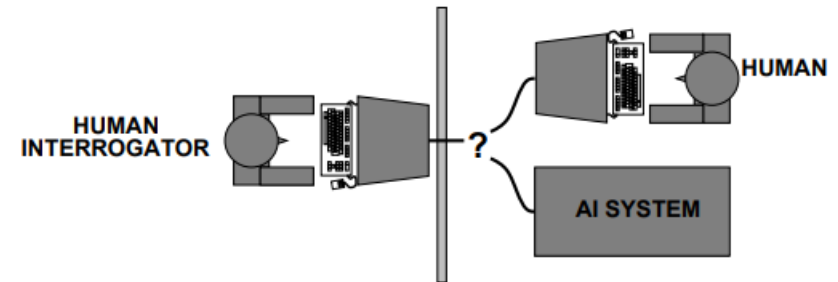
Judge tries to tell which is which.

Numerous variants

Loebner prize

Current programs nowhere close to passing this

- <http://www.jabberwacky.com/>
- <http://turingtrade.org/>



What is the Turing Test in Artificial Intelligence?

NLP to communicate successfully.

Knowledge Representation to act as its memory.

Automated Reasoning to use the stored information to answer questions and draw new conclusions.

Machine Learning to detect patterns and adapt to new circumstances.

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

Turing Test on unsuspecting judges

It is possible to (temporarily) fool humans who do not realize they may be talking to a bot

ELIZA program rephrases partner's statements and questions (~psychotherapist)

Modern variant: AOLiza

[fragment from

<http://archives.cnn.com/2000/TECH/computing/08/29/aoliza.idg/>]

five: just asking if richard can tell me about some good lobster diving spots thats it if this is a problem then forget it i dont know whats going on here

AOLiza: that is interesting. please continue.

five: continue what just ask richard when he gets home about some lobster diving up hat way thats all if you got a problem with it then forget it, this isnt worth it i dont know whats been going on or what bev has told you but i havent done anything to anybody iam the one that got used and dumped

AOLiza: tell me more about that.

five: about what

AOLiza: what do you think?

five: theres nothing to tell she used me then got tired of me said i wasnt her type anymore i was possessive and controlling

AOLiza: perhaps i already know you were possessive and controlling.

five: i was huh you nuts i gave her all the room she wanted never told her where what to do or where to go

Acting humanly: The Turing Test approach

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

The computer would need to possess the following capabilities:

1. **natural language processing** to enable it to communicate successfully in English
2. **knowledge representation** to store what it knows or hears
3. **automated reasoning** to use the stored information to answer questions and to draw new conclusions
4. **machine learning** to adapt to new circumstances and to detect and extrapolate patterns
5. **TOTAL TURING TEST**- 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

- introspection—trying to catch our own thoughts as they go by
- psychological experiments—observing a person in action
- brain imaging—observing the brain in action
- cognitive science brings together computer models from AI and experimental techniques from psychology to construct precise and testable theories of the human mind.

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

Thinking rationally: The “laws of thought” approach

SYLLOGISM: an instance of a form of reasoning in which a conclusion is drawn from two given or assumed propositions, “Socrates is a man; all men are mortal; therefore, Socrates is mortal.”

LOGIC: study of laws of thought to govern the operation of the mind

not easy to take informal knowledge and state it in the formal terms required by logical notation

Even problems with just a few hundred facts can exhaust the computational resources of any computer unless it has some guidance as to which reasoning steps to try first.

THOUGHT

BEHAVIOUR

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

HUMAN

RATIONAL

Acting rationally: The rational agent approach

An agent is just something that acts

Rational behavior is doing the right thing

Right thing is expected to maximize goal achievement, given available information

computer agents

- operate autonomously,
- perceive their environment,
- persist over a prolonged time period,
- adapt to change, and
- create and pursue goals

Rational agent is one that acts so as to achieve the best outcome or, when there is uncertainty, the **best expected outcome**

correct inference is not all of rationality in some situations, there is no provably correct thing to do, but something must still be done. There are also ways of acting rationally that cannot be said to involve inference

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

REQUIREMENTS

NATURAL LANGUAGE PROCESSING

To enable it to communicate successfully

KNOWLEDGE REPRESENTATION

Knowledge representation to store what it knows or hears;

AUTMATED REASONING

Automated reasoning to use the stored information to answer questions and to draw new conclusions

MACHINE LEARNING

machine learning to adapt to new circumstances and to detect and extrapolate patterns.

COMPUTER VISION

Computer vision to perceive objects

ROBOTICS

Robotics to manipulate objects and move about

How do we measure if Artificial Intelligence is acting like a human?

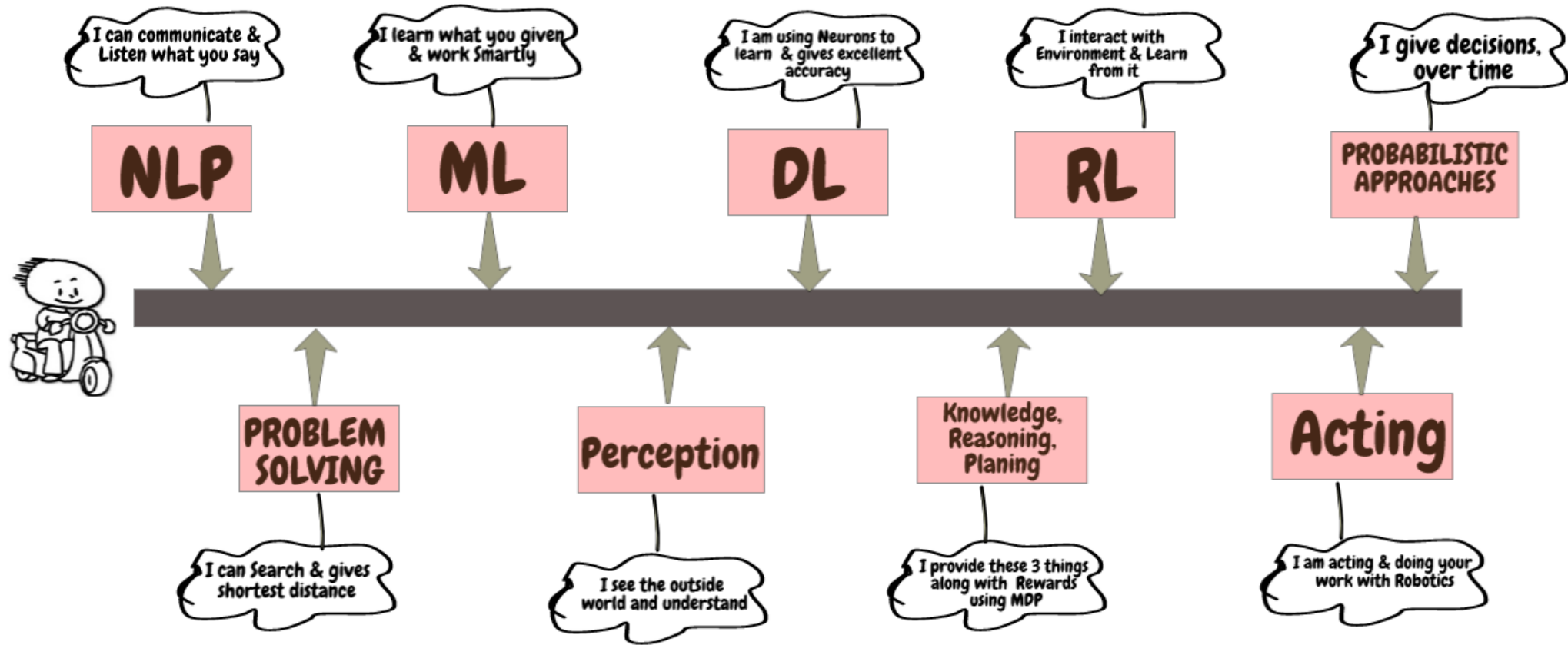
- Turing Test
- The Cognitive Modelling Approach
- The Law of Thought Approach
- The Rational Agent Approach



Artificial Intelligence

- An intelligent entity created by humans.
- Capable of performing tasks intelligently without being explicitly instructed.
- Capable of thinking and acting rationally and humanely.

Fields in AI



THE FOUNDATIONS OF ARTIFICIAL INTELLIGENCE

Philosophy	logic, methods of reasoning mind as physical system foundations of learning, language, rationality
Mathematics	formal representation and proof algorithms computation, (un)decidability, (in)tractability probability
Psychology	adaptation phenomena of perception and motor control experimental techniques (psychophysics, etc.)
Linguistics	knowledge representation grammar
Neuroscience	physical substrate for mental activity
Control theory	homeostatic systems, stability simple optimal agent designs

Philosophy

- ❑ 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?

Rationalism: power of reasoning in understanding the world

Dualism: there is a part of the human mind (or soul or spirit) that is outside of nature, exempt from physical laws

Materialism: brain's operation according to the laws of physics *constitutes* the mind

Empiricism:

Induction: general rules are acquired by exposure to repeated associations between their elements

Logical positivism: doctrine holds that all knowledge can be characterized by logical theories connected, ultimately, to **observation sentences** that correspond to sensory inputs; thus logical positivism combines rationalism and empiricism

confirmation theory: attempted to analyze the acquisition of knowledge from experience

Mathematics

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

three fundamental areas:

1. logic,
2. computation, and
3. probability.

George Boole: worked out the details of propositional, or Boolean, logic

Gottlob Frege: creating the **first order logic** that is used today

Euclid's algorithm: first nontrivial **algorithm**

Kurt Gödel: **incompleteness theorem**

Alan Turing: characterize exactly which functions *are* **computable**. Turing machine

Tractability: problem is called intractable if the time required to solve instances of the problem grows exponentially with the size of the instances

- **NP-completeness**

Despite the increasing speed of computers, careful use of resources will characterize intelligent systems

Theory of **probability:** deal with uncertain measurements and incomplete theories.

Economics

- 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?

studying how people make choices that lead to preferred outcomes

Decision theory: combines probability theory with utility theory, provides a formal and complete framework for decisions made under uncertainty

Game theory: Von Neumann and Morgenstern, a rational agent should adopt policies that are (or least appear to be) randomized. game theory does not offer an unambiguous prescription for selecting actions

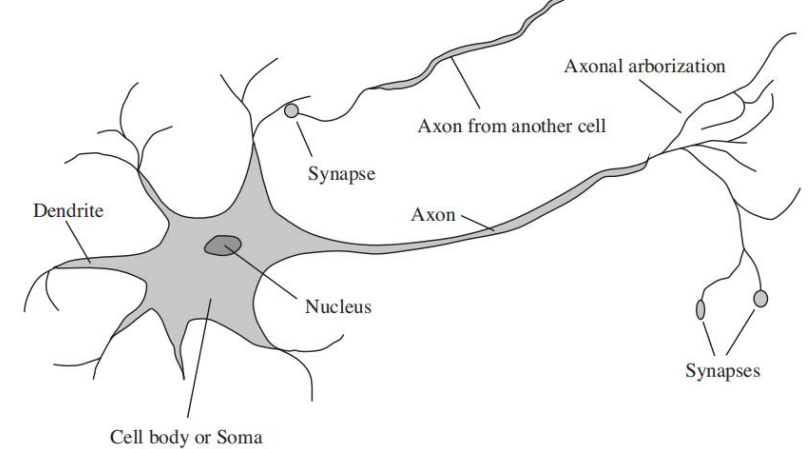
Neuroscience

- How do brains process information?

Neuroscience is the study of the nervous system, particularly the brain

Aristotle wrote, “Of all the animals, man has the largest brain in proportion to his size.”

Nicolas Rashevsky: the first to apply mathematical models to the study of the nervous system.



	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}

Figure 1.3 A crude comparison of the raw computational resources available to the IBM BLUE GENE supercomputer, a typical personal computer of 2008, and the human brain. The brain's numbers are essentially fixed, whereas the supercomputer's numbers have been increasing by a factor of 10 every 5 years or so, allowing it to achieve rough parity with the brain. The personal computer lags behind on all metrics except cycle time.

Psychology

- How do humans and animals think and act?

Computer Engineering

- How can we build an efficient computer?

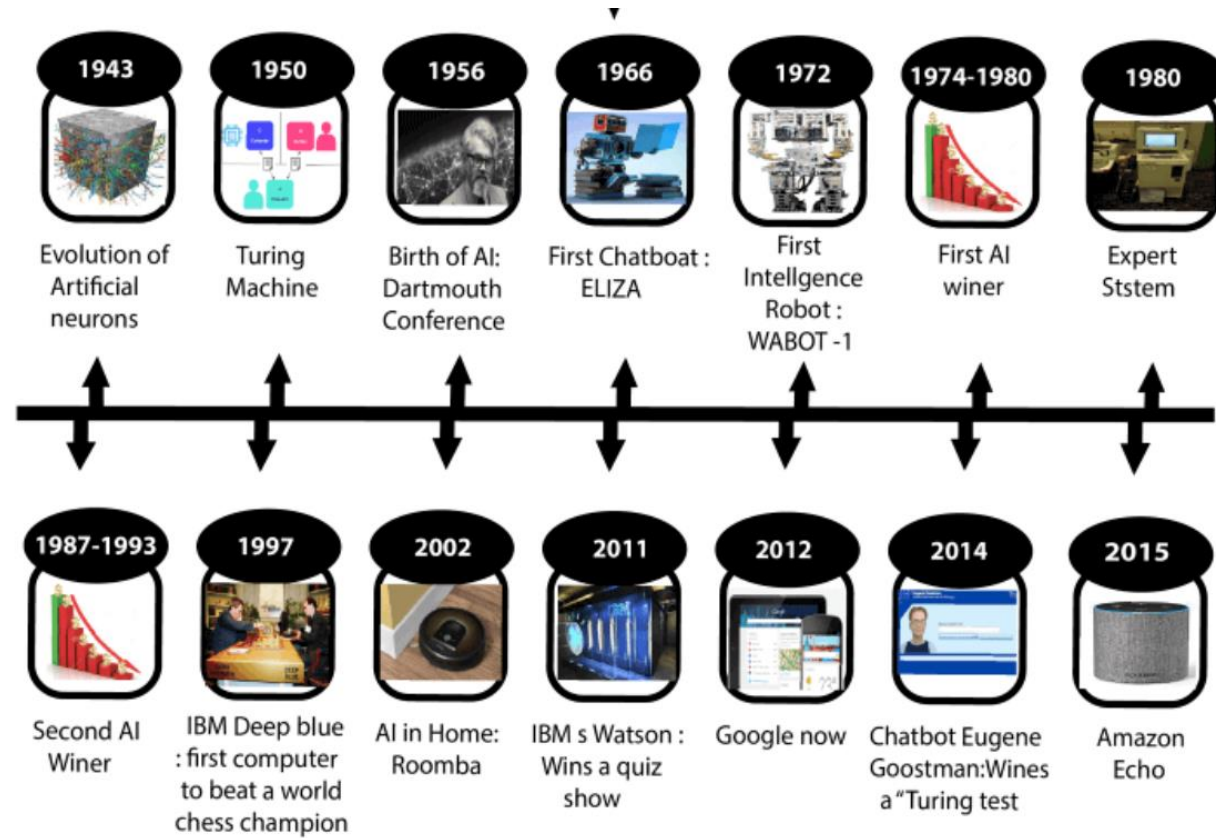
Control theory and cybernetics

- How can artifacts operate under their own control?

Linguistics

How does language relate to thought?

History of AI- Tutorial 1



Founding Fathers

Ivan Pavlov (1849-1936)

Santiago R.Y. Cajal (1879-1936)

Donal Hebb (1904-1985)

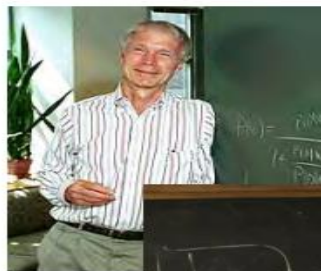
Alan L. Hodgking (1914-1998)

Andrew Huxley (1917-2012)

Marvin L. Minsky (1927-2016)

Seymour Papert (1928-2016)

Frank Rosenblatt (1928-2017)



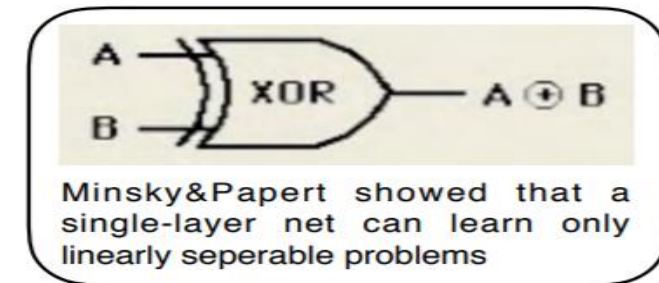
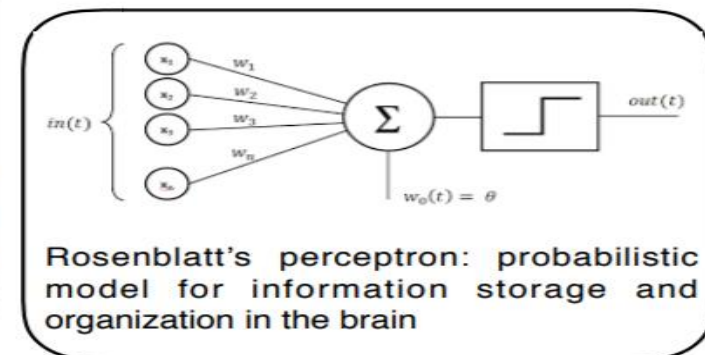
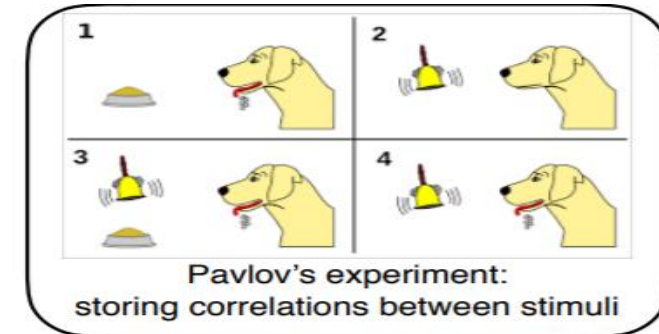
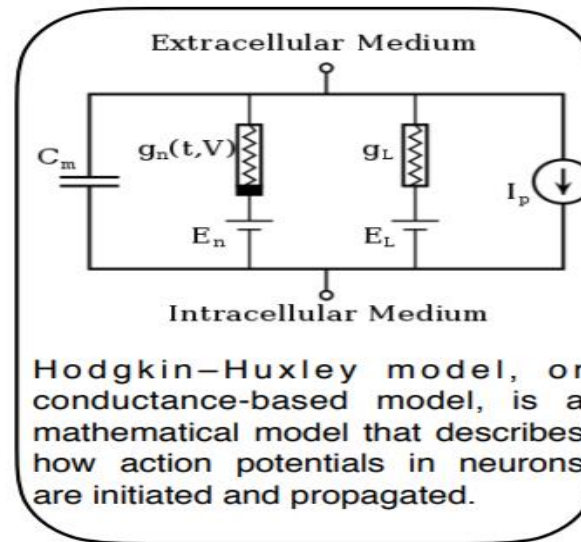
John Hopfield



Giorgio Parisi



Geoffrey Hinton



Gestation of Artificial Intelligence (1943-1955)

- **Year 1943:** The first work which is now recognized as AI was done by Warren McCulloch and Walter Pitts in 1943. They proposed a model of **artificial neurons**.
 - **Year 1949:** Donald Hebb demonstrated an updating rule for modifying the connection strength between neurons. His rule is now called **Hebbian learning**.
 - **Year 1950:** The Alan Turing who was an English mathematician and pioneered Machine learning in 1950. Alan Turing publishes "**Computing Machinery and Intelligence**" in which he proposed a test. The test can check the machine's ability to exhibit intelligent behavior equivalent to human intelligence, called a **Turing test**.
- Year 1955:** Allen Newell and Herbert A. Simon created the "first artificial intelligence program" which was named as "**Logic Theorist**". This program had proved 38 of 52 Mathematics theorems, and found new and more elegant proofs for some theorems.

The birth of artificial intelligence (1956)

The word "Artificial Intelligence" first adopted by American Computer scientist John McCarthy at the Dartmouth Conference. For the first time, AI coined as an academic field.

The golden years-Early enthusiasm (1956-1974)

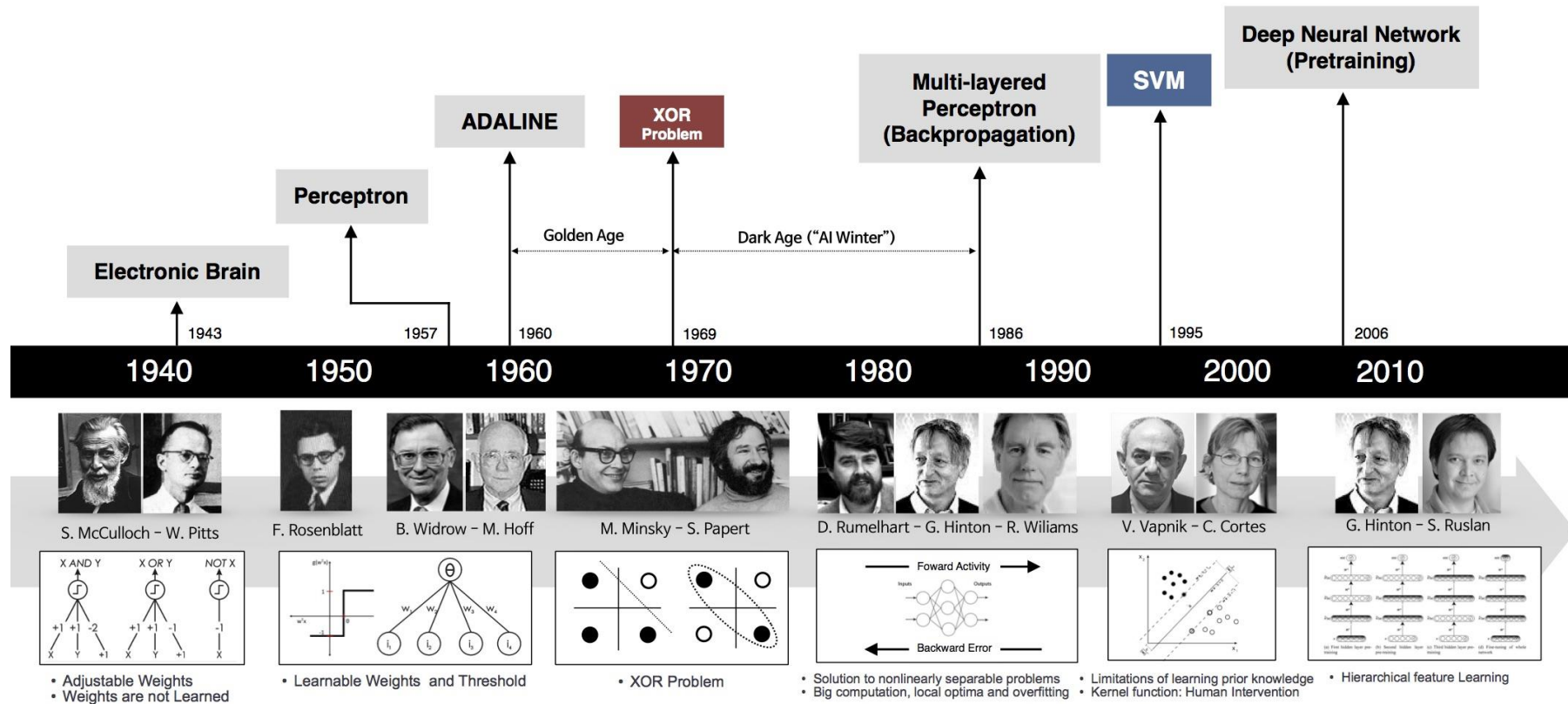
- **Year 1966:**

- The researchers emphasized developing algorithms which can solve mathematical problems. Joseph Weizenbaum created the first chatbot in 1966, which was named as ELIZA.

- **Year 1972:**

- The first intelligent humanoid robot was built in Japan which was named as WABOT-1.

Deep Networks-A Brief history



Problem Characteristics- Add on Topic- CO1 PO1

Is the problem Decomposable?

Can solution steps be ignored or undone?

Is the universe predictable?

Is a good solution Absolute or relative

Is the solution a state or path?

What is the Role of Knowledge

Does the task require interaction with a person?

Is the problem Decomposable?

Decomposable problems

Non Decomposable problems

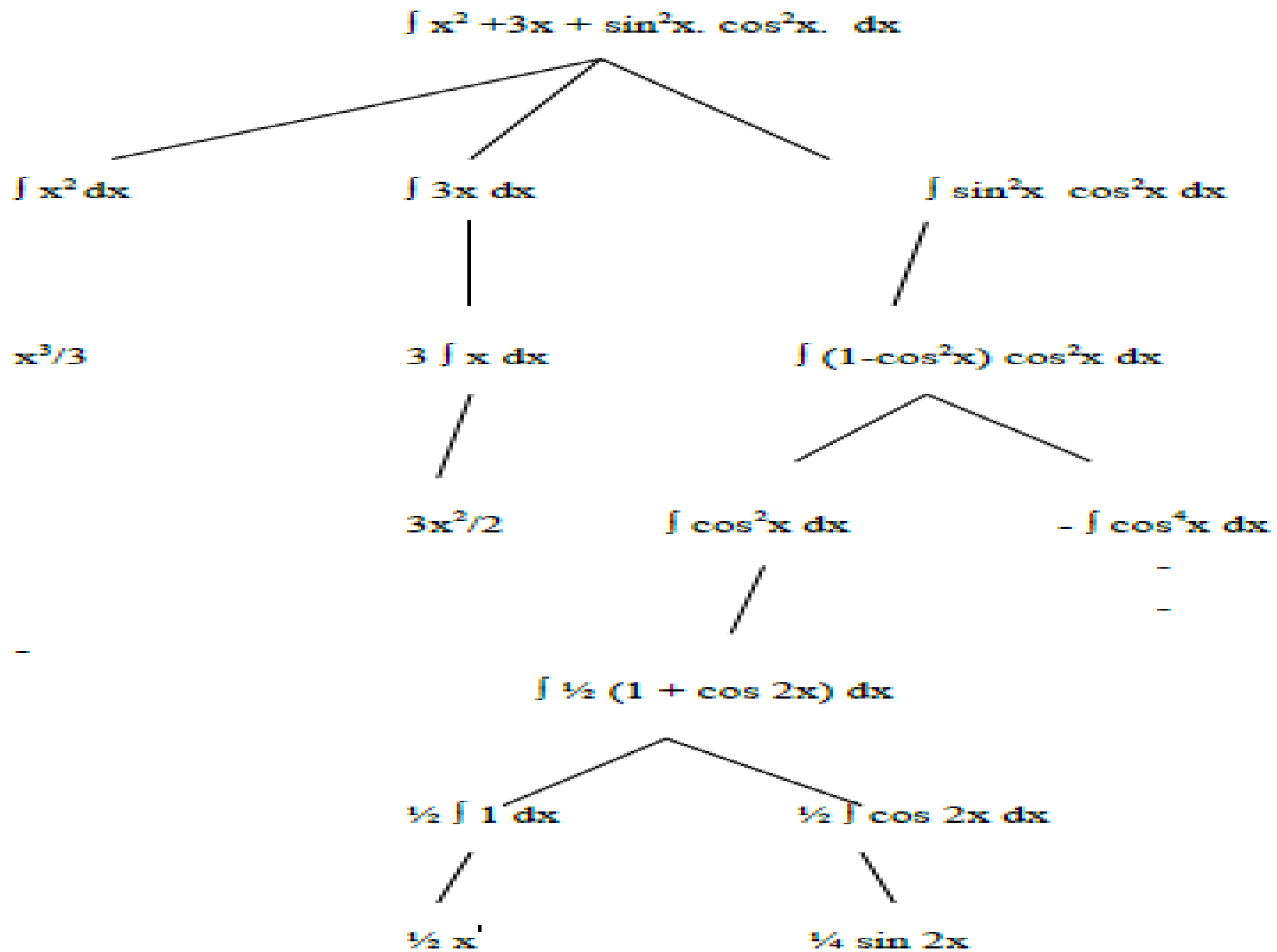
1. Decomposable problems

can solve this problem by breaking it down into three smaller problems

each of which we can then solve by using a small collection of specific rules.

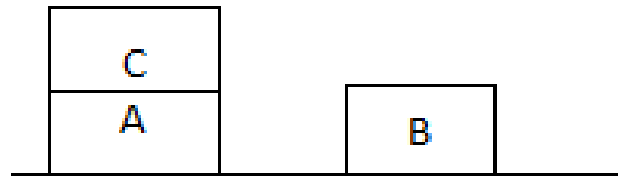
problem decomposition

$$\int (x^2 + 3x + \sin 2x \cdot \cos 2x) dx$$



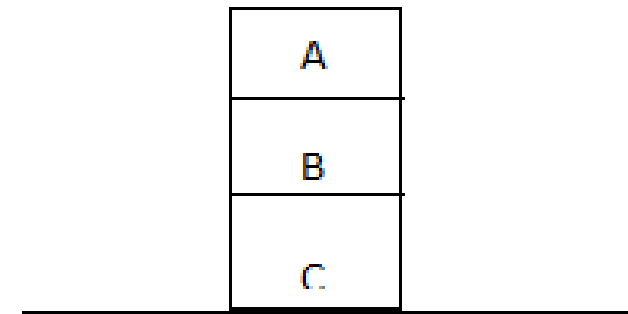
2. Non Decomposable problems

A simple blocks w Start:



ON (C,A)

Goal:



ON(B,C) and ON(A,B)

Assume that the following operators are available

1. CLEAR(x) \longrightarrow ON(x,table)
[block x has nothing on it] [pick up x and put it on the table]
2. CLEAR(x) and CLEAR(y) \longrightarrow ON(x,y)

Cont.

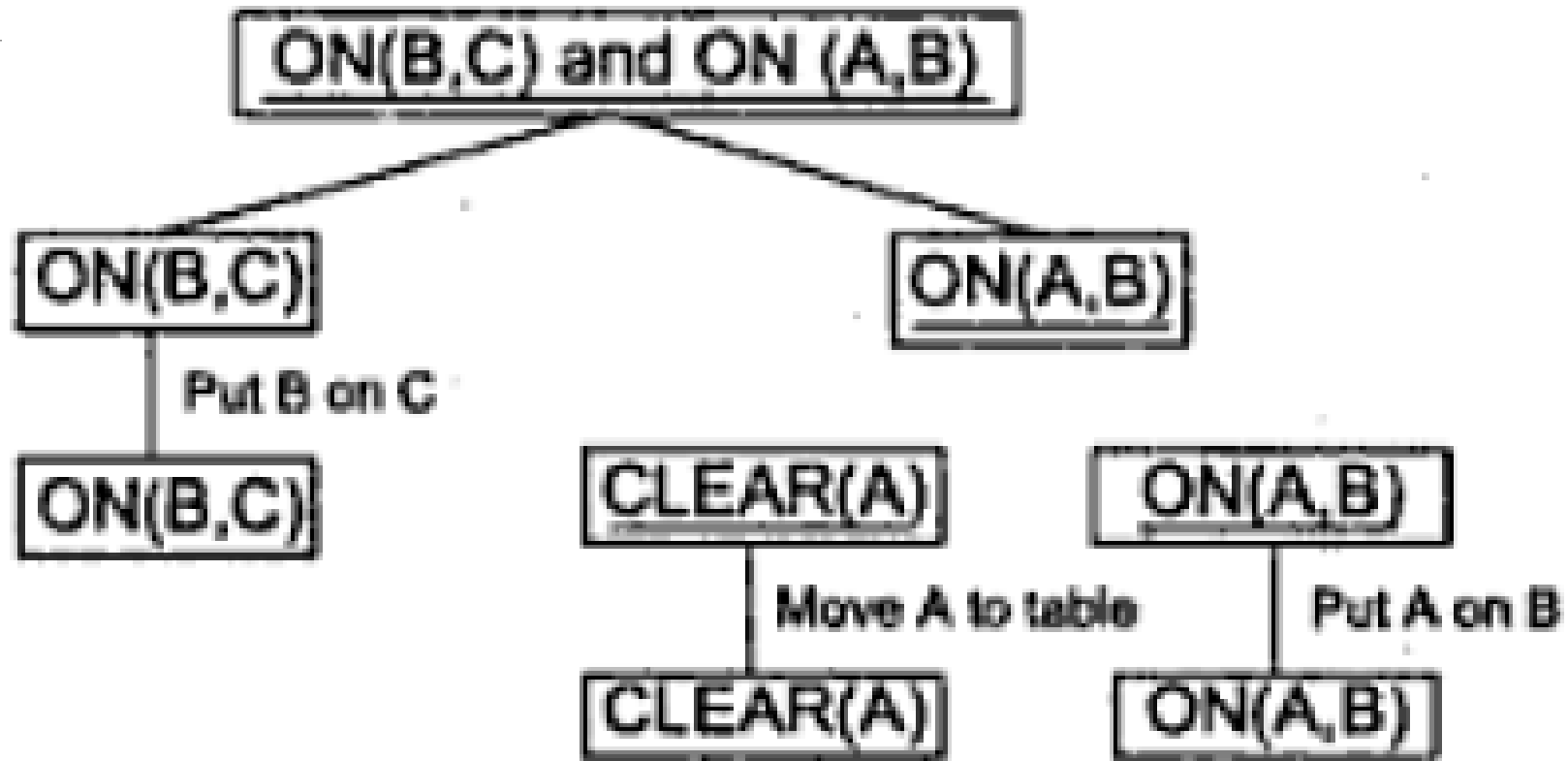


Fig. 2.11 *A Proposed Solution for the Blocks Problem*

Cont..

Regardless of which one we do first we will not be able to do the second as we had planned.

In this problem the two sub problems are not independent.

They interact and those interactions must be considered in order to arrive at a solution for entire problem.

Can solution steps be ignored or undone?

Here we can divide problems into 3 classes.

- **Ignorable**, in which solution steps can be ignored.
- **Recoverable**, in which solution steps can be undone.
- **Irrecoverable**, in which solution steps cannot be undone.

Ignorable Problem

eg, Theorem Proving

Suppose we are trying to prove a mathematical theorem.

We proceed by first proving a lemma that we think will be useful.

Eventually we realize that the lemma is no help at all.

Here the different steps in proving the theorem can be ignored.

Then we can start from another rule.

The former can be ignored.

Recoverable Problems

eg, 8 Puzzle

8-puzzle solver can keep track of the order in which operations are performed so that the operations can be undone one at a time if necessary.

7	2	4
5		6
8	3	1

Start state

	1	2
3	4	5
6	7	8

Goal state

1	2	5
3	4	
6	7	8

Irrecoverable problems

eg. Chess

Suppose a chess playing program makes a stupid move and realizes it a couple of moves later.

It cannot simply play as though it had never made the stupid move.

Nor can it simply back up and start the game over from that point.

All it can do is to **try to make the best of the current situation and go from there.**

Cont..

Ignorable problems can be solved using a simple control structure.

Recoverable problems can be solved by a slightly more complicated control strategy that does sometimes makes mistakes.

Irrecoverable problems will need to be solved by a system that expends a great deal of effort making each decision since the decision must be final.

Is the universe predictable?

Certain outcome problems

Uncertain outcome problems

Certain outcome problems

8 puzzle problem.

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

This means that it is possible to plan an entire sequence of moves and be confident that we know what the resulting state will be.

Uncertain outcome problems

Bridge

- planning may not be possible.
- One of the decisions we will have to make is which card to play on the first trick.
- it is not possible to do such planning with certainty since we cannot know exactly where all the cards are or what the other players will do on their turns.



4. Is a good solution Absolute or relative

Any Path Problem

Best Path Problem

Any path problems

Is a good solution Absolute or relative

Any path problems

- 1. Marcus was a man.
- 2. Marcus was a Pompean.
- 3. Marcus was born in 40 A. D.
- 4. all men are mortal.
- 5. All pompeans died when the volcano erupted in 79 A. D.
- 6. No mortal lives longer than 150 years.
- 7. It is now 1991 A. D.

Suppose we ask the question. “Is Marcus alive?”.

	Solutions	Axiom
1	Marcus was a man.	1
4	All men are mortal.	4
3	Marcus was born in 40 A.D.	3
7	It is now 2017 A. D.	7
9	Marcus' age is 1977 years.	3,7
6	no mortal lives longer than 150 years.	6
10	Marcus is dead.	8,6,9

Best path problems

Traveling salesman problem

Best path problems are computationally harder than any path problems.

Any path problem can often be solved in a reasonable amount of time by using heuristics that suggest good path to explore.

5. Is the solution a state or path?

Problems whose solution is a state of the world. eg. Natural language understanding. eg,

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

- Since all we are interested in is the answer to the question, it does not matter which path we follow.

Problems whose solution is a **path to a state**?

- Eg. Water jug problem
- In water jug problem, it is not sufficient to report that we have solved the problem and that the final state is (2,0).
- For this kind of problem, what we really must report is not the final state, but the path that we found to that state.

6. What is the Role of Knowledge

Problems for which a lot of knowledge is important only to constrain the search for a solution.

- Eg. Chess
- Just the rules for determining the legal moves and some simple control mechanism that implements an appropriate search procedure

Problems for which a lot of knowledge is required even to be able to recognize a solution.

- Eg. News paper story understanding

7. Does the task require interaction with a person?

Solitary problems

- Here the computer is given a problem description and produces an answer with no intermediate communication and with no demand for an explanation for the reasoning process.
- Consider the problem of proving mathematical theorems. If
 - All we want is to know that there is a proof.
 - The program is capable of finding a proof by itself.
- Then it does not matter what strategy the program takes to find the proof.

Cont..

Conversational problems

- In which there is **intermediate communication between a person and the computer**, either to provide additional assistance to the computer or to provide additional information to the user.
 - Eg. Suppose we are trying to prove some new, very difficult theorem.
 - Then the program may not know where to start.
 - At the moment, people are still better at doing the high level strategy required for a proof.
 - So the computer might like to be able to ask for advice.
 - To exploit such advice, the computer's reasoning must be analogous to that of its human advisor, at least on a few levels.

The State of the Art- What can AI do today?

Robotic vehicles

Stanley is an **autonomous car** created by **Stanford University's Stanford Racing Team** in cooperation with the **Volkswagen Electronics Research Laboratory (ERL)**. It won the 2005 DARPA Grand Challenge, earning the Stanford Racing Team a \$2 million prize.



Speech recognition

A traveler calling United Airlines to book a flight can have the entire conversation guided by an automated speech recognition and dialog management system

Autonomous planning and scheduling

NASA's Remote Agent program became the first on-board autonomous planning program to control the scheduling of operations for a spacecraft. REMOTE AGENT generated plans from high-level goals specified from the ground and monitored the execution of those plans—detecting, diagnosing, and recovering from problems as they occurred

Game playing

IBM's DEEP BLUE became the first computer program to defeat the world champion in a chess match when it bested Garry Kasparov by a score of 3.5 to 2.5 in an exhibition match



Deep Blue

Chess computer

Deep Blue was a chess-playing expert system run on a unique purpose-built IBM supercomputer. It was the first computer to win a game, and the first to win a match, against a reigning world champion under regular time controls. Development began in 1985 at Carnegie Mellon University under the name ChipTest.

Spam fighting: learning algorithms classify over a billion messages as spam, saving the recipient from having to waste time deleting what, for many users, could comprise 80% or 90% of all messages, if not classified away by algorithms

Logistics planning: During the Persian Gulf crisis of 1991, U.S. forces deployed a Dynamic Analysis and Replanning Tool, DART (Cross and Walker, 1994), to do automated logistics planning and scheduling for transportation. This involved up to 50,000 vehicles, cargo, and people at a time, and had to account for starting points, destinations, routes, and conflict resolution among all parameters. The AI planning techniques generated in hours a plan that would have taken weeks with older methods. The Defense Advanced Research Project Agency (DARPA) stated that this single application more than paid back DARPA's 30-year investment in AI

Robotics: : The iRobot Corporation has sold over two million Roomba robotic vacuum cleaners for home use. The company also deploys the more rugged PackBot to Iraq and Afghanistan, where it is used to handle hazardous materials, clear explosives, and identify the location of snipers

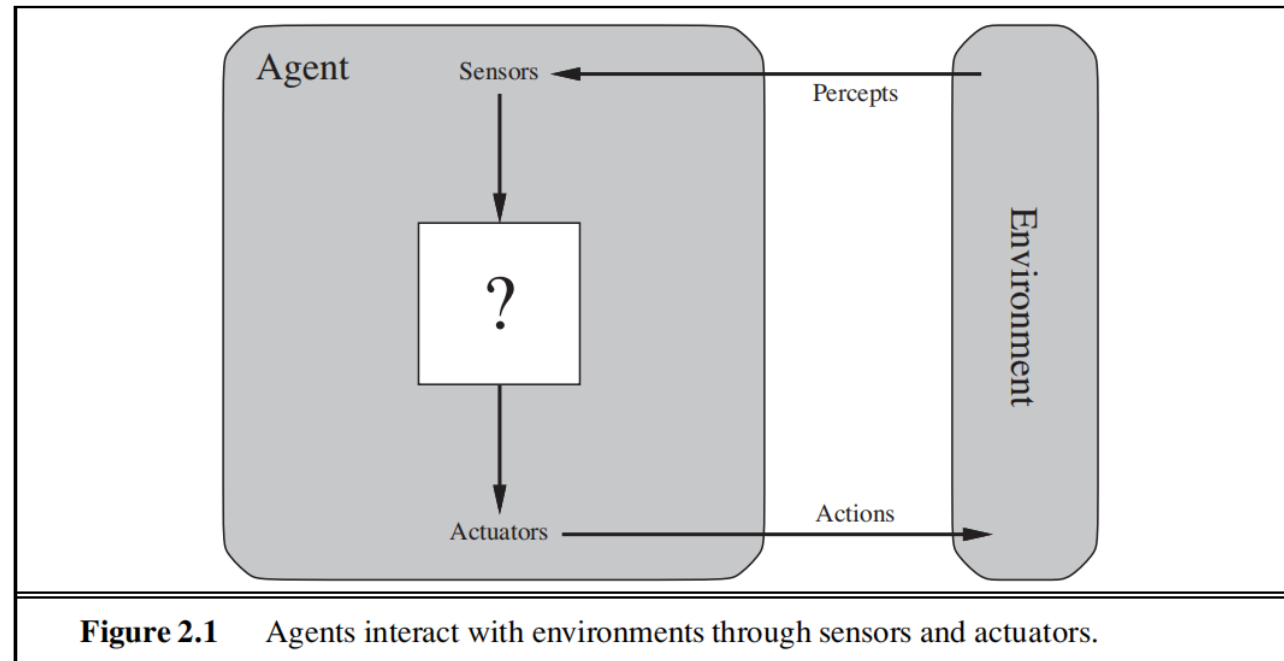


Machine Translation: A computer program automatically translates from Arabic to English, allowing an English speaker to see the headline “Ardogan Confirms That Turkey Would Not Accept Any Pressure, Urging Them to Recognize Cyprus.” The program uses a statistical model built from examples of Arabic-to-English translations and from examples of English text totaling two trillion words. None of the computer scientists on the team speak Arabic, but they do understand statistics and machine learning algorithms.

INTELLIGENT AGENTS

Agents and Environments

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



the term percept to refer to the agent's perceptual inputs at any given instant

An agent's percept sequence is the complete history of everything the agent has ever perceived

an agent's choice of action at any given instant can depend on the entire percept sequence observed to date, but not on anything it hasn't perceived

an agent's behavior is described by the **agent function** that maps any given percept sequence to an action

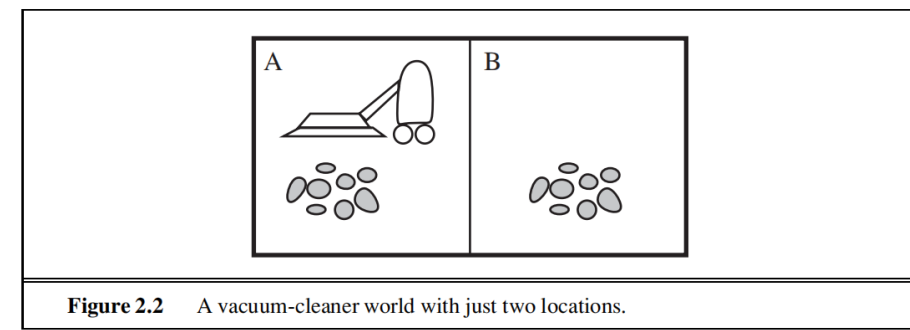
$$[f: P^* \rightarrow A]$$

The **agent program** runs on the physical **architecture** to produce f , **the agent function for an artificial agent will be implemented by an agent program**

agent = architecture + program

The agent function is an abstract mathematical description; the agent program is a concrete implementation, running within some physical system.

The Vacuum Cleaner World



This particular world has just two locations: squares A and B.

The vacuum agent perceives which square it is in and whether there is dirt in the square.

It can choose to move left, move right, suck up the dirt, or do nothing.

One very simple agent function is the following: if the current square is dirty, then suck; otherwise, move to the other square.

Percepts: location and contents, e.g., [A,Dirty]

Actions: *Left, Right, Suck, NoOp*

Agent's function → *look-up table*

For many agents this is a very large table

Percept sequence	Action
<i>[A, Clean]</i>	<i>Right</i>
<i>[A, Dirty]</i>	<i>Suck</i>
<i>[B, Clean]</i>	<i>Left</i>
<i>[B, Dirty]</i>	<i>Suck</i>
<i>[A, Clean], [A, Clean]</i>	<i>Right</i>
<i>[A, Clean], [A, Dirty]</i>	<i>Suck</i>
<i>⋮</i>	<i>⋮</i>
<i>[A, Clean], [A, Clean], [A, Clean]</i>	<i>Right</i>
<i>[A, Clean], [A, Clean], [A, Dirty]</i>	<i>Suck</i>
<i>⋮</i>	<i>⋮</i>

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

function REFLEX-VACUUM-AGENT(*[location, status]*) **returns** an action

if *status = Dirty* **then return** *Suck*
else if *location = A* **then return** *Right*
else if *location = B* **then return** *Left*

Figure 2.8 The agent program for a simple reflex agent in the two-state vacuum environment. This program implements the agent function tabulated in Figure 2.3.

Good Behaviour: Concept of Rationality

- A **rational agent** is one that does the right thing
- what does it mean to do the right thing?
 - by considering the *consequences* of the agent's behavior

When an agent is plunked down in an environment, it generates a sequence of actions according to the percepts it receives. This sequence of actions causes the environment to go through a sequence of states. If the sequence is desirable, then the agent has performed well.

This notion of desirability is captured by a **performance measure** that evaluates any given sequence of environment states.

Example- Vacuum Cleaner Revisited

We might propose to measure performance by the amount of dirt cleaned up in a single eight-hour shift.

With a rational agent, of course, what you ask for is what you get.

A rational agent can maximize this performance measure by cleaning up the dirt, then dumping it all on the floor, then cleaning it up again, and so on.

A more suitable performance measure would reward the agent for having a clean floor.

For example, one point could be awarded for each clean square at each time step (perhaps with a penalty for electricity consumed and noise generated).

As a general rule, it is better to design performance measures according to what one actually wants in the environment, rather than according to how one thinks the agent should behave.

Performance measure: *An objective criterion for success of an agent's behavior.*

Performance measures of a vacuum-cleaner agent: amount of dirt cleaned up, amount of time taken, amount of electricity consumed, level of noise generated, etc.

Performance measures self-driving car: time to reach destination (minimize), safety, predictability of behavior for other agents, reliability, etc.

Performance measure of game-playing agent: win/loss percentage (maximize), robustness, unpredictability (to “confuse” opponent), etc.

Rationality

- What is rational at any given time depends on four things:
 - Performance measuring success
 - Agents prior knowledge of environment
 - Actions that agent can perform
 - Agent's percept sequence to date
- **Rational Agent:** For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

Example- Vacuum Cleaner

- The performance measure awards one point for each clean square at each time step
- The “geography” of the environment is known a priori but the dirt distribution and the initial location of the agent are not. Clean squares stay clean and sucking cleans the current square. The Left and Right actions move the agent left and right except when this would take the agent outside the environment, in which case the agent remains where it is.
- The only available actions are Left, Right, and Suck.
- The agent correctly perceives its location and whether that location contains dirt

Rational Agent

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

Omniscience, learning, and autonomy

Omniscient agent: knows the *actual* outcome of its actions and can act accordingly; but omniscience is

impossible in reality.

rationality is not the same as perfection.

Rationality maximizes *expected* performance, while perfection maximizes *actual* performance

information gathering

Exploration

learn as much as possible from what it perceives

rational agent should be **autonomous**—it should learn what it can to compensate for partial or incorrect prior knowledge

Task Environment PEAS

PEAS (Performance, Environment, Actuators, Sensors)

In designing an agent, the first step must always be to specify the task environment as fully as possible

Example: the task of designing a **self-driving car**

- **Performance measure** Safe, fast, legal, comfortable trip, maximize profits
- **Environment** Roads, other traffic, pedestrians, customers
- **Actuators** Steering wheel, accelerator, brake, signal, horn
- **Sensors** Cameras, LIDAR (light/radar), speedometer, GPS, odometer, engine sensors, keyboard

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.

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display of scene categorization	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, suggestions, corrections	Keyboard entry
Figure 2.5 Examples of agent types and their PEAS descriptions.				

Task Environment Types

- Fully observable (vs. partially observable)
- Single agent (vs. Multi agent)
- Deterministic (vs. stochastic)
- Episodic (vs. sequential)
- Static (vs. dynamic)
- Discrete (vs. continuous)
- Known (vs. unknown)

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 we say that the task environment is fully observable.

A task environment is effectively fully observable if the sensors detect all aspects that are *relevant* to the choice of action; relevance, in turn, depends on the performance measure.

Fully observable environments are convenient because the agent need not maintain any internal state to keep track of the world.

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

If the agent has no sensors at all then the environment is **unobservable**

Single agent vs. multiagent

If only one agent is involved in an environment, and operating by itself then such an environment is called single agent environment.

However, if multiple agents are operating in an environment, then such an environment is called a multi-agent environment.

chess is a competitive multiagent environment.

In the taxi-driving environment avoiding collisions maximizes the performance measure of all agents, so it is a partially cooperative multiagent environment.

- It is also partially competitive because, for example, only one car can occupy a parking space.

Deterministic vs. stochastic

If the next state of the environment is completely determined by the current state and the action executed by the agent, then we say the environment is deterministic; otherwise, it is stochastic.

an agent need not worry about uncertainty in a fully observable, deterministic environment.

If the environment is partially observable, however, then it could *appear* to be stochastic.

environment is **uncertain** if it is not fully observable or not deterministic.

“stochastic” generally implies that uncertainty about outcomes is quantified in terms of probabilities
a nondeterministic environment is one in which actions are characterized by their *possible* outcomes,
but no probabilities are attached to them

Episodic vs. sequential

Episodic task environment:

- the agent's experience is divided into atomic episodes.
- In each episode the agent receives a percept and then performs a single action.
- Crucially, the next episode does not depend on the actions taken in previous episodes.
- Many classification tasks are episodic.

Sequential environments

- the current decision could affect all future decisions
- Chess and taxi driving are sequential: in both cases, short-term actions can have long-term consequences.
- Episodic environments are much simpler than sequential environments because the agent does not need to think ahead.

Static vs. dynamic

If the environment can change while an agent is deliberating, then we say the environment is dynamic for that agent; otherwise, it is static

A static environment does not change while the agent is thinking.

The passage of time as an agent deliberates is irrelevant.

Dynamic environments, on the other hand, are continuously asking the agent what it wants to do; if it hasn't decided yet, that counts as deciding to do nothing.

If the environment itself does not change with the passage of time but the agent's performance score does, then we say the environment is **semi-dynamic**.

Chess, when played with a clock, is semi-dynamic.

Crossword puzzles are static

Discrete vs. continuous

If the number of distinct percepts and actions is limited, the environment is discrete, otherwise it is continuous.

The chess environment has a finite number of distinct states (excluding the clock).

Chess also has a discrete set of percepts and actions.

Taxi driving is a continuous-state and continuous-time problem: the speed and location of the taxi and of the other vehicles sweep through a range of continuous values and do so smoothly over time

Taxi-driving actions are also continuous (steering angles, etc.).

Input from digital cameras is discrete, strictly speaking, but is typically treated as representing continuously varying intensities and locations.

Known vs. unknown

In a known environment, the outcomes (or outcome probabilities if the environment is stochastic) for all actions are given.

If the environment is unknown, the agent will have to learn how it works in order to make good decisions

a *known* environment can be *partially* observable

- for example, in solitaire card games, I know the rules but am still unable to see the cards that have not yet been turned over.

An *unknown* environment can be *fully* observable

- in a new video game, the screen may show the entire game state but I still don't know what the buttons do until I try them.

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

Figure 2.6 Examples of task environments and their characteristics.

Structure of Agent

The job of AI is to design an **agent program** that implements the agent function the mapping from percepts to actions.

his program will run on some sort of computing device with physical sensors and actuators called the **architecture**

$$\textit{agent} = \textit{architecture} + \textit{program}$$

architecture makes the percepts from the sensors available to the program, runs the program, and feeds the program's action choices to the actuators as they are generated

Agent programs

Agent program: use current percept as input from the sensors and return an action to the actuators

Agent function: takes the entire percept history

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

  append percept to the end of percepts
  action  $\leftarrow$  LOOKUP(percepts, table)
  return action
```

Figure 2.7 The TABLE-DRIVEN-AGENT program is invoked for each new percept and returns an action each time. It retains the complete percept sequence in memory.

To build a rational agent in this way, we as designers must construct a table that contains the appropriate action for every possible percept sequence.

Let \mathcal{P} be the set of possible percepts and let T be the lifetime of the agent (the total number of percepts it will receive)

The lookup table will contain $\sum_{t=1}^T |\mathcal{P}|^t$ entries

Consider the automated taxi: the visual input from a single camera comes in at the rate of roughly 27 megabytes per second (30 frames per second, 640×480 pixels with 24 bits of color information). This gives a lookup table with over 10250,000,000,000 entries for an hour's driving.

Even the lookup table for chess a tiny, well-behaved fragment of the real world would have at least 10150 entries.

The daunting size of these tables (the number of atoms in the observable universe is less than 10^{80}) means that

- a) no physical agent in this universe will have the space to store the table,
- b) the designer would not have time to create the table,
- c) no agent could ever learn all the right table entries from its experience, and
- d) even if the environment is simple enough to yield a feasible table size, the designer still has no guidance about how to fill in the table entries.

Types of Agent Programs

Four basic kinds of agent programs that embody the principles underlying almost all intelligent systems:

1. Simple reflex agents;
2. Model-based reflex agents;
3. Goal-based agents; and
4. Utility-based agents

Simple reflex agents

Select actions on the basis of the current percept, ignoring the rest of the percept history

Agents **do not have memory** of past world states or percepts.

So, actions depend solely on **current percept**.

Action becomes a “reflex.”

Uses **condition-action rules**.

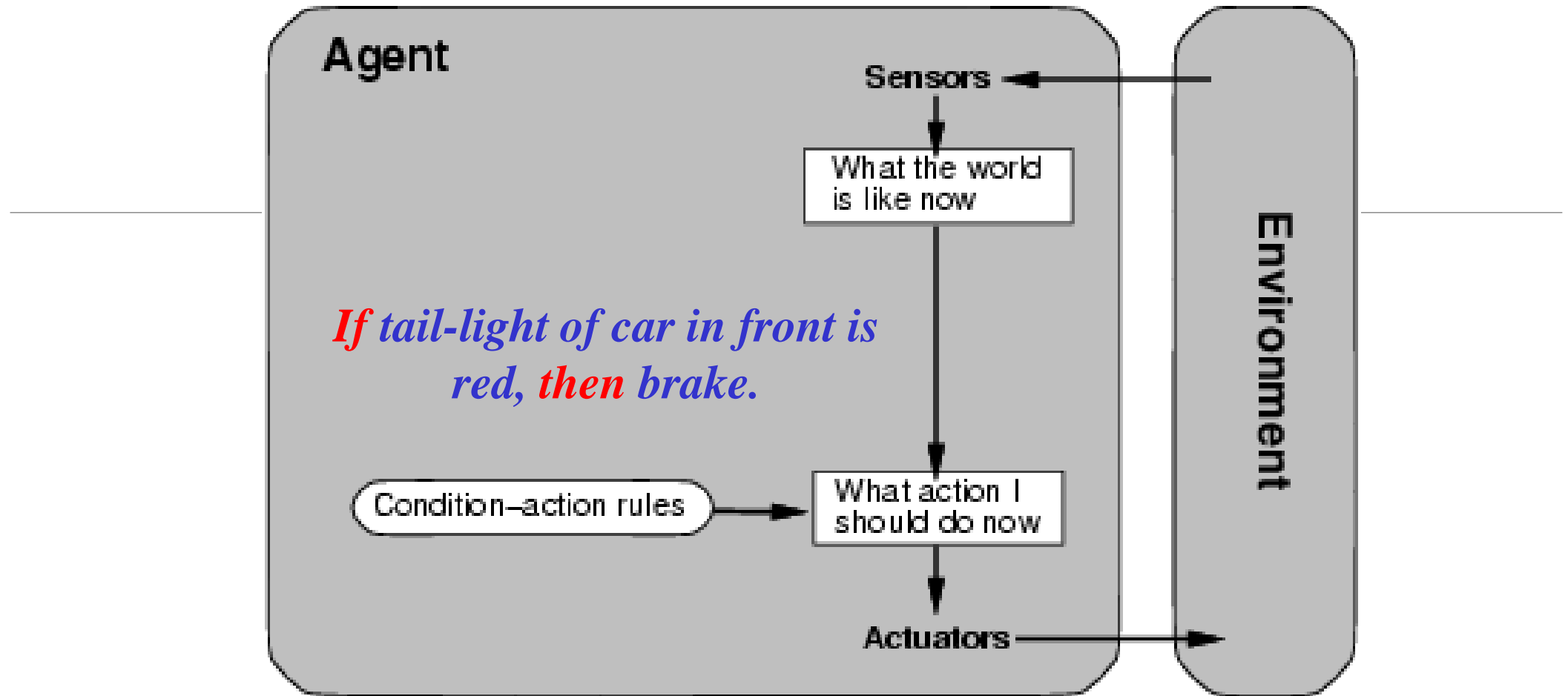
function REFLEX-VACUUM-AGENT(*[location, status]*) **returns** an action

if *status* = *Dirty* **then return** *Suck*
else if *location* = *A* **then return** *Right*
else if *location* = *B* **then return** *Left*

Figure 2.8 The agent program for a simple reflex agent in the two-state vacuum environment. This program implements the agent function tabulated in Figure 2.3.

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.



condition-action rule

if *car-in-front-is-braking* **then** *initiate-braking*

function SIMPLE-REFLEX-AGENT(*percept*) **returns** an action

persistent: *rules*, a set of condition–action rules

state \leftarrow INTERPRET-INPUT(*percept*)

rule \leftarrow RULE-MATCH(*state*, *rules*)

action \leftarrow *rule*.ACTION

return *action*

Figure 2.10 A simple reflex agent. It acts according to a rule whose condition matches the current state, as defined by the percept.

The INTERPRET-INPUT function generates an abstracted description of the current state from the percept, and

the RULE-MATCH function returns the first rule in the set of rules that matches the given state description. Note that the description in terms of “rules” and “matching” is purely conceptual;

actual implementations can be as simple as a collection of logic gates implementing a Boolean circuit

This will work *only if the correct decision can be made on the basis of only the current percept—that is, only if the environment is fully observable.*

Even a little bit of unobservability can cause serious trouble. For example, the braking rule given earlier assumes that the condition *car-in-front-is-braking* can be determined from the current percept—a single frame of video.

This works if the car in front has a centrally mounted brake light.

Infinite loops are often unavoidable for simple reflex agents operating in partially observable environments

Escape from infinite loops is possible if the agent can randomize its actions.

Model-based reflex agents

Key difference (wrt simple reflex agents):

- Agents have **internal state**, which is used to keep track of past states of the world.
- Agents have the ability **to represent change in the World**.

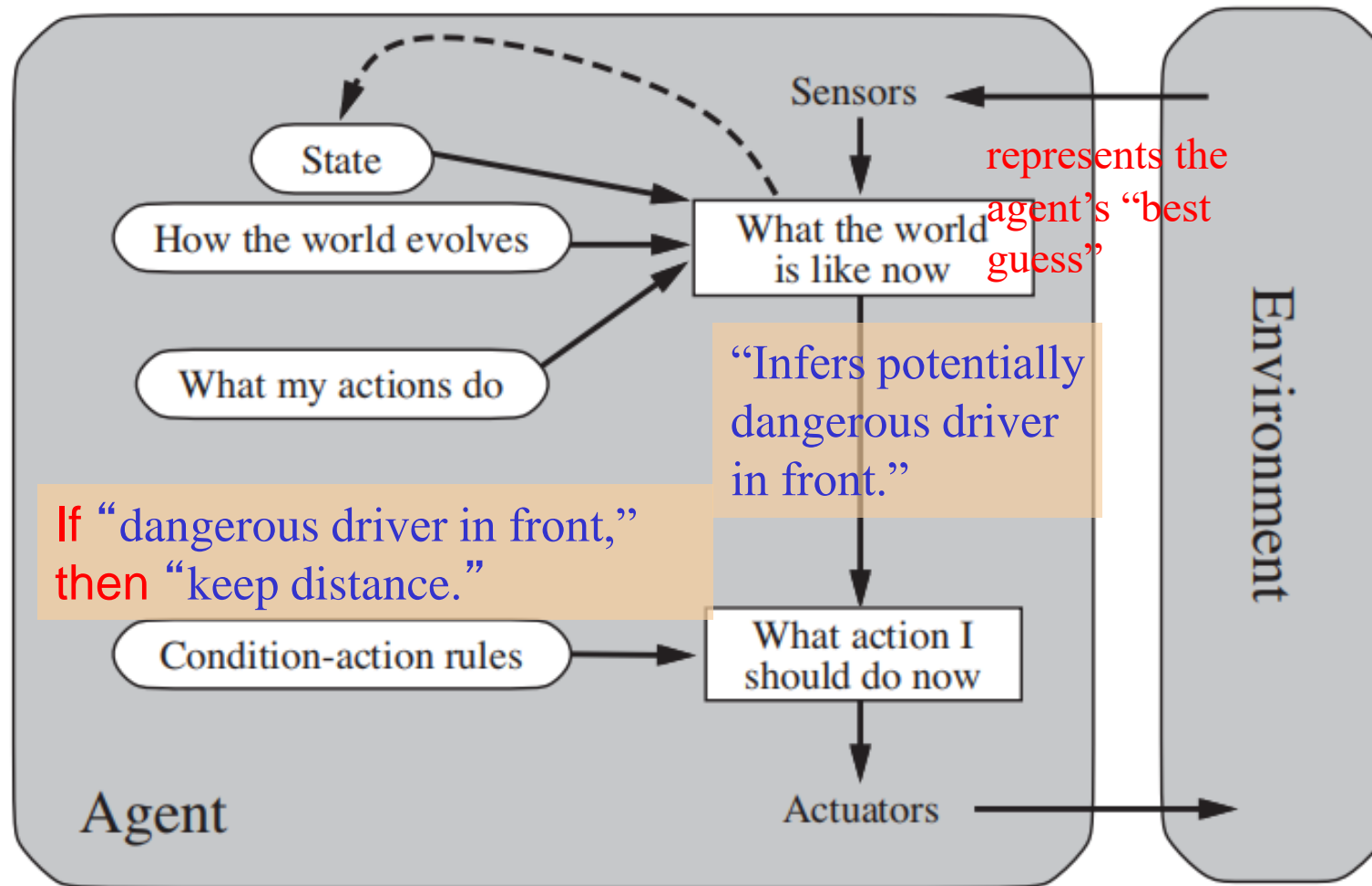


Figure 2.11 A model-based reflex agent.

internal state information as time goes by requires two kinds of knowledge to be encoded in the agent program

1. we need some information about how the world evolves independently of the agent
2. we need some information about how the agent's own actions affect the world

knowledge about “how the world works is called a **model** of the world. An agent that uses such a model is called a **model-based agent**.

```
function MODEL-BASED-REFLEX-AGENT(percept) returns an action
  persistent: state, the agent's current conception of the world state
               model, a description of how the next state depends on current state and action
               rules, a set of condition–action rules
               action, the most recent action, initially none

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

Figure 2.12 A model-based reflex agent. It keeps track of the current state of the world, using an internal model. It then chooses an action in the same way as the reflex agent.

UPDATE-STATE, which is responsible for creating the new internal state description.

Goal-based agents

Key difference wrt Model-Based Agents:

In addition to state information, have **goal information** that **describes desirable situations to be achieved**.

Search and **planning** are the subfields of AI devoted to finding action sequences that achieve the agent's goals

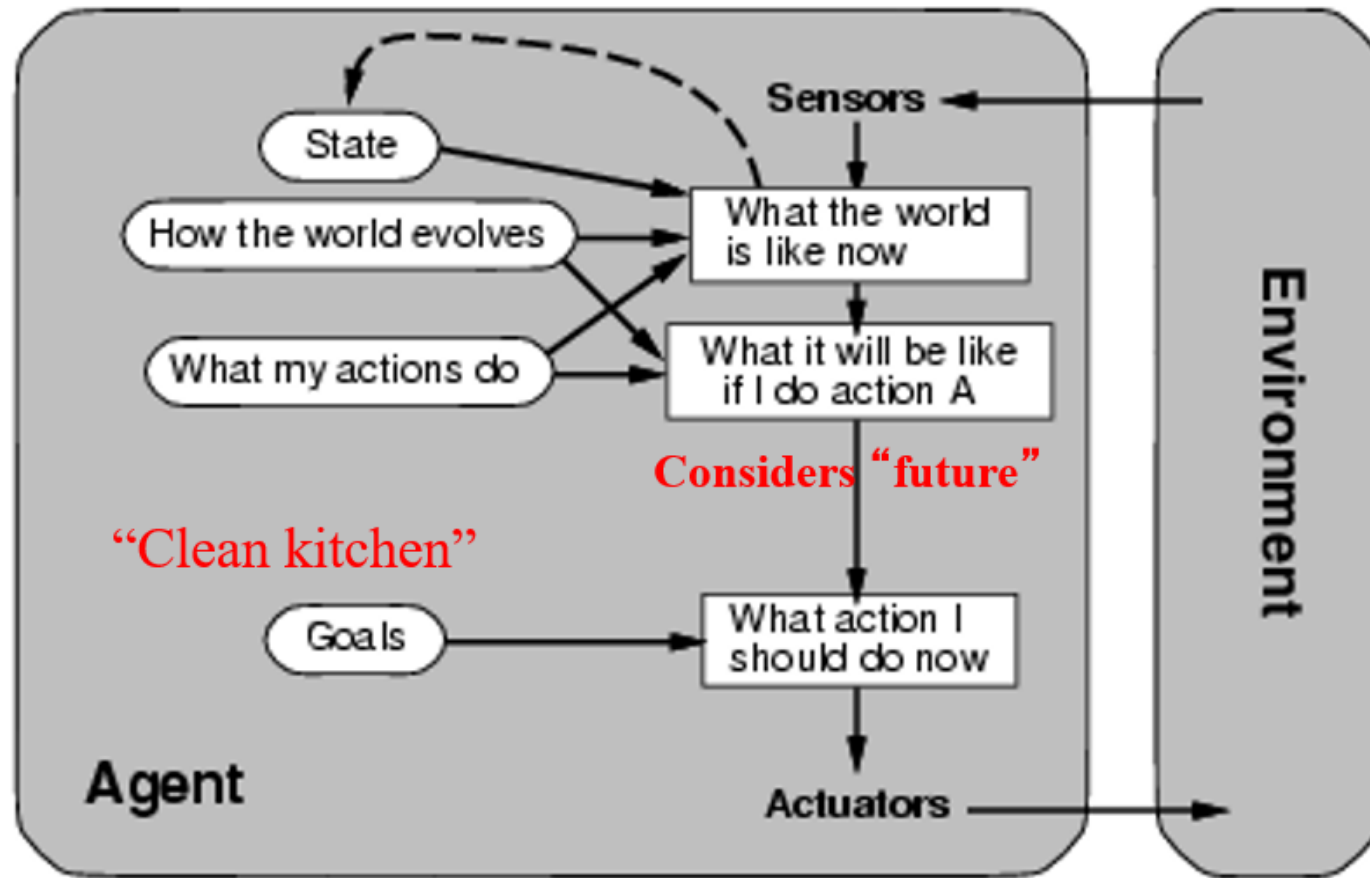
Agents of this kind take **future** events into consideration.

What *sequence* of actions can I take to achieve certain goals?

Choose actions so as to (eventually) achieve a (given or computed) goal.

**Module:
Problem Solving**

Goal-based agents



Agent keeps track of the world state as well as set of goals it's trying to achieve: chooses actions that will (eventually) lead to the goal(s).

More flexible than reflex agents → may involve **search and planning**

Utility-based agents

Goals alone are not enough to generate high-quality behavior in most environments

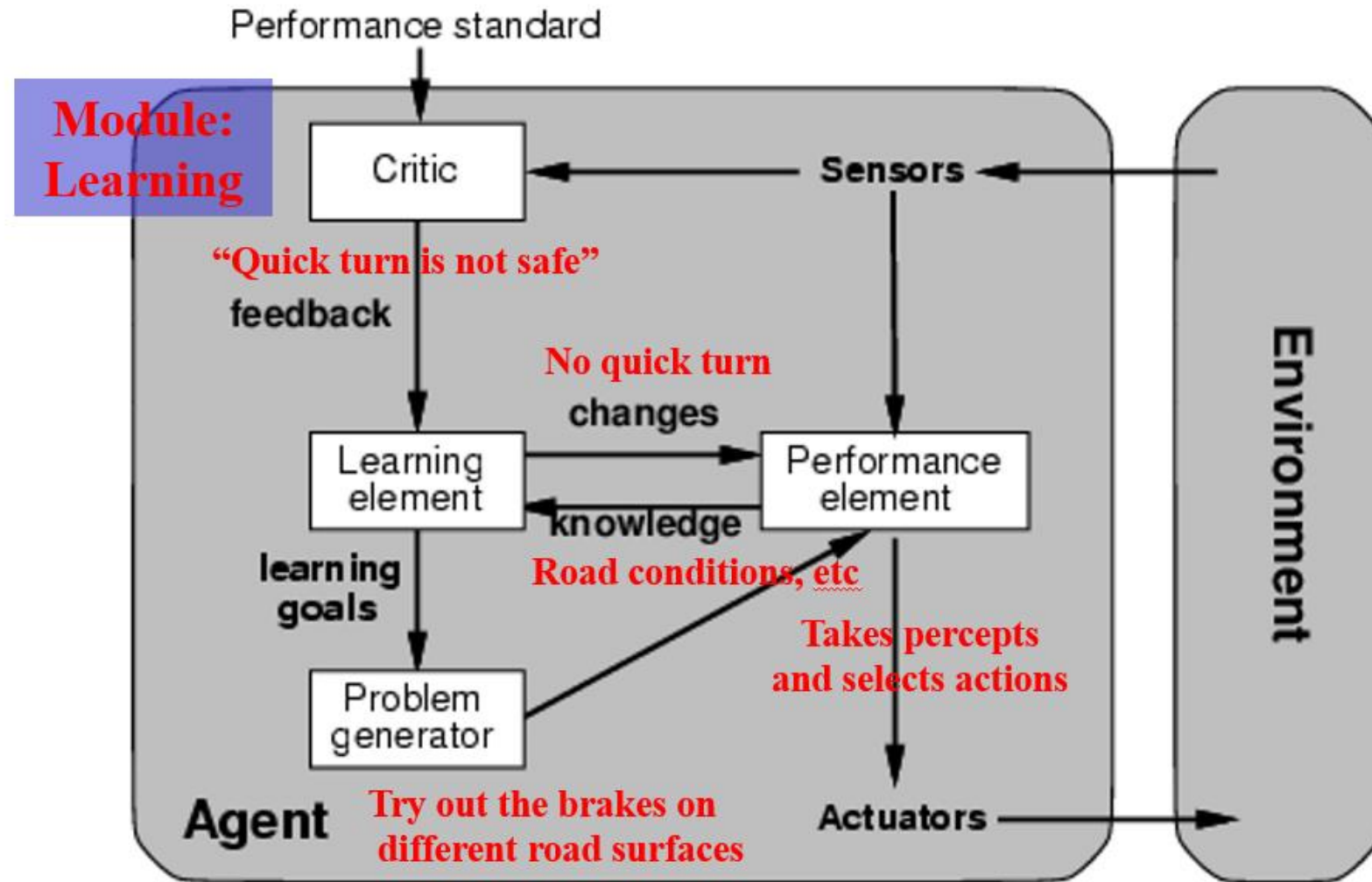
Goals just provide a crude binary distinction between “happy” and “unhappy” states

Because “happy” does not sound very scientific, economists and computer scientists use the term **utility** instead

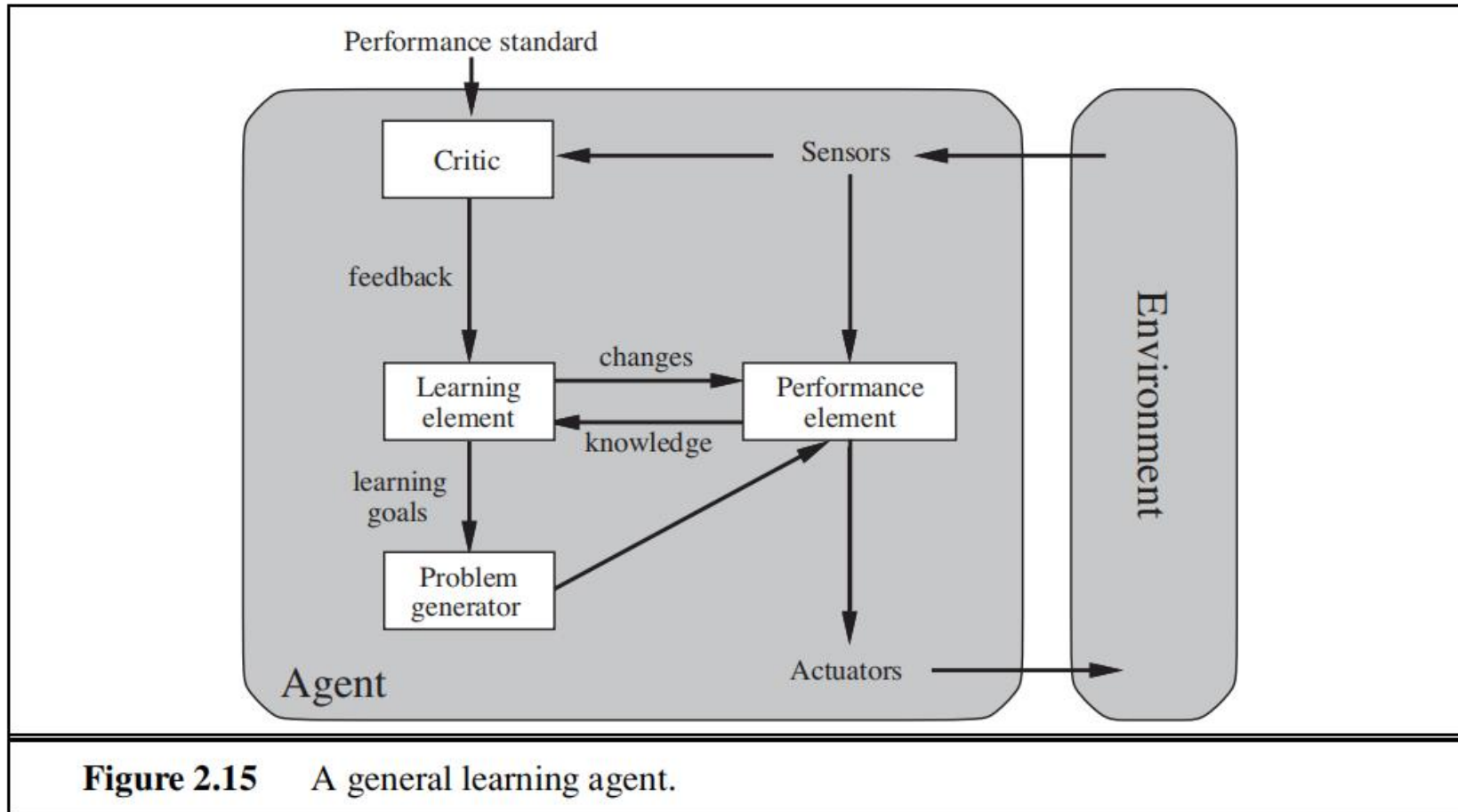
An agent’s **utility function** is essentially an internalization of the performance measure. If the internal utility function and the external performance measure are in agreement, then an agent that chooses actions to maximize its utility will be rational according to the external performance measure.

More complicated when agent needs to learn
utility information: Reinforcement learning
(based on action payoff)

Learning agents
Adapt and improve over time



Learning agents



A learning agent can be divided into four conceptual components

1. **learning element**, which is responsible for making improvements
2. **performance element**, which is responsible for selecting external actions. The performance element is what we have previously considered to be the entire agent: it takes in percepts and decides on actions.
3. **Feedback from the critic** : on how the agent is doing and determines how the performance element should be modified to do better in the future
4. **problem generator**. It is responsible for suggesting actions that will lead to new and informative experiences



Thank You!

MODULE 1 ENDS

THANK YOU