Artificial Intelligence

Topics

- Defining Artificial Intelligence.
- Foundations of AI,
- Applications of AI.
- **Intelligent agents:** Agents and Environments.
- Structure of agents.

Introduction to AI

- A branch of Computer Science named Artificial Intelligence (AI) pursues creating the computers / machines as intelligent as human beings.

John McCarthy the father of Artificial Intelligence described AI as, "***The science and engineering of making intelligent machines, especially intelligent computer programs***".

Artificial Intelligence (AI) is a branch of Science which deals with helping machines find solutions to complex problems in a more human-like fashion.

- This generally involves borrowing characteristics from human intelligence, and applying them as algorithms in a computer friendly way.
- A more or less flexible or efficient approach can be taken depending on the requirements established, which influences how artificial the intelligent behaviour appears.

- AI is one of the newest fields in science and engineering. Work started in earnest soon after World War II, and the name itself was coined in 1956.
- AI currently encompasses a huge variety of subfields, ranging from the general (learning and perception) to the specific, such as *playing chess, proving mathematical theorems, writing poetry, driving a car on a crowded street, and diagnosing diseases.*

Defining AI techniques.

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

Thinking Humanly	Thinking Rationally
"The exciting new effort to make computers think . . . *machines with minds*, in the full and literal sense." (Haugeland, 1985) "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . ." (Bellman, 1978)	"The study of mental faculties through the use of computational models." (Charniak and McDermott, 1985) "The study of the computations that make it possible to perceive, reason, and act." (Winston, 1992)
Acting Humanly	**Acting Rationally**
"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)	"Computational Intelligence is the study of the design of intelligent agents." (Poole *et al.*, 1998) "AI . . . is concerned with intelligent behavior in artifacts." (Nilsson, 1998)

Figure 1.1 Some definitions of artificial intelligence, organized into four categories.

Thinking Humanly:
Cognitive science- modeling the processes of human thought.

- A computer program to be considered as thinking like a human, we must understand how humans think.
- There are three main ways to do this:
  - **through introspection** (examining our own thoughts),
  - **psychological experiments** (observing people in action), and
  - **brain imagi**ng (observing the brain's activity).
- We can create computer programs based on this understanding.
- If a program's behavior matches human behavior, it suggests that some of the mechanisms in the program might also be operating in humans.
- Allen Newell and Herbert Simon, who developed the **General Problem Solver (GPS)**, and how they compared the program's reasoning steps with human problem-solving processes.
- This interdisciplinary approach, combining AI and psychology, is called cognitive science.

Acting Humanly: The Turing test approach

- How do you distinguish intelligent behavior from intelligence?
- Turing test, by A. Turing, 1950: determining if a program qualifies as artificially intelligent by subjecting it to an interrogation along with a human counterpart.
- The program passes the test if a human judge cannot distinguish between the answers of the program and the answers of the human subject.

- To pass the Turing Test, a computer would need several key capabilities:
- **Natural Language Processing (NLP)**: This enables the computer to understand and communicate effectively in natural languages like English. It allows the computer to interpret written questions and generate appropriate responses.
- **Knowledge Representation**: The computer must be able to store and organize information it acquires from various sources, such as text or speech. This stored knowledge forms the basis for the computer's responses to questions.

- **Automated Reasoning**: The computer should be capable of using the stored information to answer questions and draw new conclusions logically. This involves processing the available data to arrive at valid responses.
- **Machine Learning**: The computer needs the ability to adapt to new situations and learn from experience. Machine learning algorithms enable the computer to detect patterns in data, make predictions, and improve its performance over time based on feedback.

Difference between Turing & Total Turing test

Turing test	Total Turing test
Involves a human interrogator communicating with both a computer and another human through written text (typically via a computer interface).	Involves not only written communication but also incorporates video signals, allowing the interrogator to test the computer's perceptual abilities (such as recognizing objects in a video feed).
The test focuses primarily on text-based communication and does not involve physical interaction between the interrogator and the computer.	Additionally, the Total Turing Test may involve physical interaction between the interrogator and the computer, symbolized by the possibility of passing physical objects
Success is achieved if the interrogator cannot reliably distinguish between the computer and the human based on their responses alone.	Success in the Total Turing Test requires the computer to not only communicate convincingly through text but also to perceive and interact with the physical world in a manner similar to humans.

- To pass the total Turing Test, the computer will need
  - COMPUTER VISION : to perceive objects, and
  - ROBOTICS : to manipulate objects and move about

Thinking Rationally: "laws of thought approach"

- In Artificial Intelligence thinking rationally means thinking rightly for example if something is true that should be true or that must be true or it can't be false
- If someone thinking rightly always in a given circumstance in a given amount of information then we can call it as laws of thought approach
- The greek philosopher Aristotle was one of the first to attempt to codify "Right Thinking" for structured argument. He always yielded a correct conclusion when given correct premises. For example

" Socrates is a men, all men are mortal, therefore Socrates is mortal"

"All men have brains, All humans have brains therefore all humans are men"

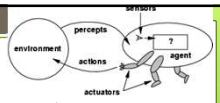
Aristotle's Contribution

- Aristotle is credited with one of the earliest attempts to systematize "right thinking" or valid reasoning processes.
- He introduced syllogisms, which are structured arguments that lead to irrefutable conclusions if the premises are true.
- Aristotle's work laid the foundation for the study of logic, which aims to understand and formalize the rules of valid inference.
- In the 19th century, logicians developed precise notation systems for expressing statements about objects and their relations.
- This notation allowed for the formal representation of logical arguments and provided a framework for reasoning about all kinds of objects and relationships.

Logic in Artificial Intelligence

- By 1965, there were programs capable of solving any problem expressible in logical notation, theoretically.
- However, there were limitations to this approach. Converting informal knowledge into formal logical statements, especially when certainty is less than 100%, is challenging.
- Additionally, even with powerful computational resources, solving problems practically can be difficult, especially when dealing with large amounts of data or complex reasoning tasks.
- Despite these challenges, the "logicist" tradition within artificial intelligence aims to build intelligent systems based on logical reasoning principles.

Acting Rationally: A Rational agent approach



- Rational behavior: doing the right thing.
- Many AI applications adopt the intelligent agent approach.
- An agent is an entity capable of generating action.
- In AI a rational agent must be autonomous, capable of perceiving its environment, adaptable, with a given goal.
- A "rational agent" is one that acts to achieve the best outcome or, when things are uncertain, the best expected outcome.
- Not all rational behavior involves logical reasoning. For instance, quickly pulling your hand away from a hot stove is a reflex action that's usually more successful than thinking about it first.
- Most often the agents are small pieces of code with a specific proficiency. The problem is solved by combining the skills of several agents.

- Acting rationally means making decisions and taking actions that aim to achieve the best possible outcome based on the available information and given the specific goals or objectives.
- In the context of artificial intelligence (AI), a rational agent is one that acts to maximize its performance measure, considering the constraints and uncertainties of its environment.

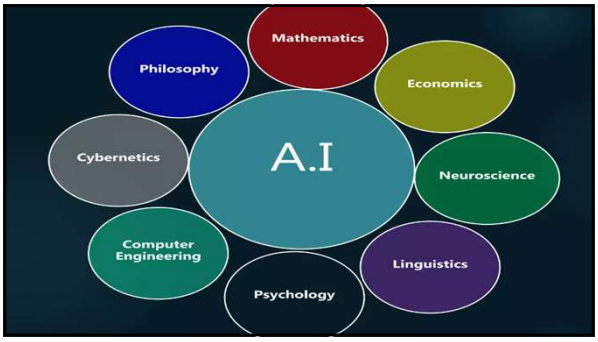
Acting humanly : The Turing Test Approach

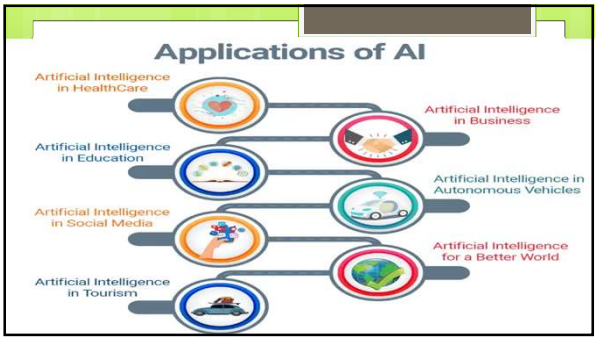
Thinking humanly : The Cognitive modeling approach

Thinking Rationally : The "laws of thought" approach

Acting Rationally : The rational agent approach

- **Thinking Humanly**: Thinking humanly involves replicating human cognitive processes, including intuition, emotions, and heuristics, in AI systems.
- **Acting Humanly**: Acting humanly means behaving in a way that mimics human social interactions, emotions, and communication.
- **Thinking Rationally**: Thinking rationally involves using logic and systematic reasoning to make decisions and solve problems.
- **Acting Rationally**: Acting rationally means making decisions and taking actions that aim to achieve the best possible outcome based on available information and goals.





- **Artificial Intelligence in Healthcare**: AI is used to improve diagnostics, personalize treatment plans, and manage patient data more efficiently.
- **Artificial Intelligence in Education**: AI aids in personalized learning experiences, automates administrative tasks, and provides intelligent tutoring systems.
- **Artificial Intelligence in Social Media**: AI is employed to analyze user behavior, curate content, and enhance user experience through personalized recommendations.

- **Artificial Intelligence in Business**: AI helps in automating routine tasks, enhancing customer service through chatbots, and making data-driven business decisions.
- **Artificial Intelligence in Autonomous Vehicles**: AI powers self-driving cars, enabling them to navigate and make decisions on the road.
- **Artificial Intelligence in Tourism**: AI enhances travel planning, customer service, and personalized recommendations for travelers.
- **Artificial Intelligence for a Better World**: AI is used for environmental monitoring, disaster response, and creating sustainable solutions to global challenges.

Intelligent Agents : Agents & Environments

- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
- Human agent:
  - eyes, ears, and other organs for sensors;
  - hands, legs, mouth, and other body parts for actuators
- Robotic agent:
  - cameras and infrared range finders for sensors
  - various motors for actuators
- Software agent:
  Receives keystrokes, file contents, and network packets as sensory inputs and acts on the environment by displaying on the screen, writing files, and sending network packets.

Continued…

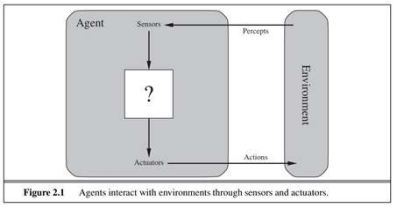
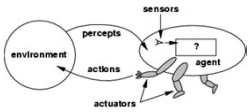


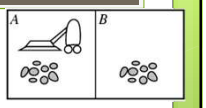
Figure 2.1 Agents interact with environments through sensors and actuators.

Continued…

- Agent's behavior is described by the agent function.
- The agent function maps from percept histories to actions:
  - [f: P* → A]
- **percept sequence** is the complete history of everything the agent has ever perceived.
- The agent function is an abstract mathematical description; the agent program is a concrete implementation, running within some physical system to produce f
- agent = architecture + program



Vacuum-cleaner world



- Percepts: location and contents, e.g., [A,Dirty]
- Actions: *Left*, *Right*, *Suck*, *NoOp*
- *Agent's function* → *look-up table*
- The table is, of course, an external characterization of the agent. Internally, the agent function for an artificial agent will be implemented by an agent program.
  - *For many agents this is a very large table*

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

PEAS

- **Performance Measure:** Performance measure is the unit used to define an agent's success. The performance of agents changes according to their distinct principles.
  - 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**
- **Actuator:** An actuator is a component of the agent that provides the action's output to the environment.
- **Sensor:** Sensors are the receptive components of an agent that receive input.

PEAS

- **Environment:** The environment is an agent's immediate surroundings. If the agent is set in motion, it changes over time. There are five primary types of environments:
  - Fully Observable & Partially Observable
  - Episodic & Sequential
  - Static & Dynamic
  - Discrete & Continuous
  - Deterministic & Stochastic

Specifying the Task Environment - PEAS

PEAS: Performance measure, Environment, Actuators, Sensors

- Must first specify the setting for intelligent agent design
- Consider, e.g., *the task of designing an automated taxi driver*:
  - ***Performance measure:*** Safe, fast, legal, comfortable trip, maximize profits
  - ***Environment:*** Roads, other traffic, pedestrians, customers, stray animals, road works, police cars, Left /right driving based on location.
  - ***Actuators:*** Steering wheel, accelerator, brake, signal, horn, display screen or voice synthesizer.
  - ***Sensors:*** Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard

Performance measure

- Getting to the correct destination;
- minimizing fuel consumption, the trip time or cost, violations of traffic laws and disturbances to other drivers;
- Maximizing profits, safety and passenger comfort.

Continued…

- **Agent:** Interactive English tutor
- **Performance measure:** Maximize student's score on test
- **Environment:** Set of students
- **Actuators:** Screen display (exercises, suggestions, corrections)
- **Sensors:** Keyboard

Environment types

- Fully observable (vs. partially observable)
- Deterministic (vs. stochastic)
- Episodic (vs. sequential)
- Static (vs. dynamic)
- Discrete (vs. continuous)
- Single agent (vs. multiagent):

- **Fully observable / Partially observable**
  - If an agent's sensors give it access to the complete state of the environment needed to choose an action, the 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;
    (e.g. chess)
  - 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.

Deterministic / Stochastic

- An environment is **deterministic** if the next state of the environment is completely determined by the current state of the environment and the action of the agent;
- In a **stochastic** environment, there are multiple, unpredictable outcomes.
- If the environment is partially observable, however, then it could appear to be stochastic..
- Most real situations are so complex that it is impossible to keep track of all the unobserved aspects; for practical purposes, they must be treated as stochastic.
- We say an environment is uncertain if it is not fully observable or not deterministic.
  - In a fully observable, deterministic environment, the agent need not deal with uncertainty.
  - "stochastic" generally implies that uncertainty about outcomes is quantified in terms of probabilities.

- Example:
- Taxi driving is clearly stochastic in this sense, because one can never predict the behavior of traffic exactly;
- moreover, one's tires blow out and one's engine seizes up without warning.
- The vacuum world as we described it is deterministic,
  - variations can include stochastic elements such as randomly appearing dirt and an unreliable suction mechanism

Non-deterministic environment

- In a nondeterministic environment, when an agent takes an action, there can be several possible outcomes, but there are no probabilities associated with these outcomes. This means that while the agent can anticipate different potential results from its actions, it cannot predict which one will occur.
- For example, if an agent moves a robot forward, the possible outcomes might include:
- The robot moves forward successfully.
- The robot gets stuck.
- The robot encounters an obstacle.
- In a nondeterministic environment, the agent knows these outcomes are possible, but it doesn't know the likelihood of each one happening.

Episodic / Sequential

- In an episodic environment, the agent's experience is divided into **atomic episodes.** Each episode consists of the agent perceiving and then performing a single action.
- Subsequent episodes do not depend on what actions occurred in previous episodes. Choice of action in each episode depends only on the episode itself.

(E.g., classifying images.)

- In a classification task like spotting defective parts on an assembly line, the agent evaluates each part independently.
- Whether a part is defective or not is based solely on the current observation and action, without any regard to previous decisions or actions.
- In a sequential environment, the agent engages in a series of connected episodes. **Current decision can affect future decisions.** (E.g., chess and driving)

Static / Dynamic

- **Static Environment**:
  - The environment does not change while the agent is deciding on an action.
  - The agent can take its time to deliberate without worrying about changes or the passage of time affecting the environment.
  - Example: **Crossword puzzles**. The puzzle remains the same regardless of how long the agent takes to decide on a move.
- **Dynamic Environment**:
  - The environment changes while the agent is deliberating.
  - The agent must continuously monitor the environment and make timely decisions.
  - If the agent takes too long to decide, it effectively counts as deciding to do nothing.
  - Example: **Taxi driving**. The traffic and the taxi's movement continue regardless of the agent's decision-making process.

- **Semi dynamic Environment**:
  - The environment itself does not change with time, but the agent's performance score may be affected by the passage of time.
  - Example: **Chess played with a clock**. The state of the chessboard does not change on its own, but the player's time to make a move is limited, impacting the player's performance score (time left).

Static: No changes in the environment while the agent decides.
Dynamic: Continuous changes in the environment, requiring prompt decisions from the agent.
Semi dynamic: The environment is static, but time constraints affect the agent's performance.

Discrete / Continuous

- If the number of distinct percepts and actions is limited, the environment is discrete, otherwise it is continuous.
- **Discrete Environment**:
  - **State**: The environment has a finite number of distinct states.
  - **Time**: Time is handled in discrete steps or intervals.
  - **Percepts and Actions**: Both are represented by a finite set of possibilities.
  - **Example**: Chess. The board has a finite number of positions (states), moves occur in turns (discrete time), and there are specific, distinct moves (discrete actions).
- **Continuous Environment**:
  - **State**: The environment has states that can take on any value within a continuous range.
  - **Time**: Time flows smoothly and continuously.
  - **Percepts and Actions**: Both can vary continuously, allowing for an infinite range of possibilities.
  - **Example**: Taxi driving. The position and speed of the taxi, as well as other vehicles, can take on any value within a range, and these values change smoothly over time. Steering angles and acceleration are also continuous actions.

Known vs. unknown

- **Known Environment**:
  - The agent (or its designer) knows the "laws of physics" or the rules that govern the environment.
  - The outcomes or probabilities of outcomes for all actions are provided or understood.
  - The agent can make decisions based on this complete knowledge.
  - **Example**: Solitaire card games. The rules are known, but some elements (e.g., the face-down cards) might be hidden, making the environment partially observable.
- **Unknown Environment**:
  - The agent (or its designer) does not know the rules or the outcomes of actions.
  - The agent must learn how the environment works through exploration and experimentation.
  - **Example**: A new video game. The player can see the entire game state (fully observable) but does not yet understand the effects of different buttons or actions until they are tried.

Single agent / Multi-agent

- **Single-Agent Environment**:
  - This involves only one agent operating in its environment. For
  - Example, a crossword puzzle solver is a single-agent scenario because the agent (puzzle solver) interacts only with the puzzle itself, without any other competing or cooperating entities.
- **Multi-Agent Environment**:
  - This involves multiple agents that can interact with each other.
  - For instance, in chess, there are two agents (players), each with their own strategies and goals. The actions of one agent (player) directly affect the performance of the other, making it a multi-agent environment.

- **Competitive vs. Cooperative**:
  - **Competitive Multi-Agent Environment**: In games like chess, each agent aims to maximize their own performance, which directly impacts the other's performance negatively. This creates a competitive dynamic.
  - **Cooperative Multi-Agent Environment**: In scenarios like driving a taxi, all agents (drivers) generally aim to avoid collisions and follow traffic rules, which benefits all. This environment can be partially competitive and partially cooperative, as there might still be competitive aspects (e.g., securing a parking spot).
- **Agent Interaction**:
  - Whether another entity is treated as an agent or merely as an object depends on whether its behavior impacts or is impacted by the agent's own performance measures.
  - For example, in chess, the opponent is an agent because their actions affect your performance and vice versa.

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable	Yes	Yes	No
Deterministic	Strategic	Strategic	No
Episodic	No	No	No
Static	Semi	Yes	No
Discrete	Yes	Yes	No
Single agent	No	No	No

The environment type largely determines the agent design

The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

THE STRUCTURE OF AGENTS

- The job of AI is to design an **agent program** that implements the agent function— the mapping from percepts to actions.
- We assume this program will run on some sort of computing device with physical sensors and actuators—we call this the **architecture.**

agent= architecture+ program

- The architecture might be just an ordinary PC, or it might be a robotic car with several onboard computers, cameras, and other sensors.
- Agent program that keeps track of the percept sequence and then uses it to index into a table of actions to decide what to do.

```
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 ← LOOKUP(percepts, table)
  return action
```

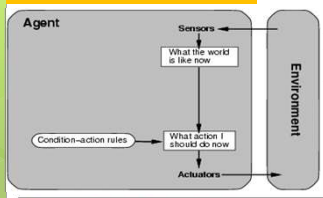
Figure 2.7 The TABLE-DRIVEN-AGENT program is invoked for each new percept and returns an action each time. It retains the complete percept sequence in memory.

- **Table-Driven Agent Function**
- **Function Name**: TABLE-DRIVEN-AGENT(percept)
  - **Purpose**: Returns an action based on the percepts (inputs from the environment).
- **Persistent Variables:**
  - **percepts**: A sequence of all percepts encountered so far, initially empty.
  - **table**: A predefined table that maps sequences of percepts to actions.
- **Procedure:**
  - **Append**: The current percept is appended to the end of the percepts sequence.
  - **Lookup**: The agent uses the LOOKUP function to find the action corresponding to the current sequence of percepts in the table.
  - **Return**: The action found in the table is returned.
- **Explanation**: The TABLE-DRIVEN-AGENT program is executed for each new percept received. It keeps a complete record of all percepts encountered in memory and returns an action based on this sequence.

Types of Agents

- Simple reflex agents
- Reflex agents with state/model
- Goal-based agents
- Utility-based agents

Simple reflex agents



- Simple but very limited intelligence.
- **Action does not depend on percept history, only on current percept.**
- Therefore no memory requirements.

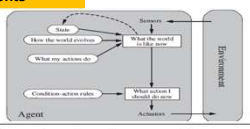
```
function REFLEX-VACUUM-AGENT([location,status]) returns an action
  if status = Dirty then return Suck
  else if location = A then return Right
  else if location = B then return Left
```

- A **simple reflex agent** is one of the most basic types of intelligent agents. It makes decisions based solely on the current percept (input) without considering any previous percepts or history.
  - **Condition-Action Rules**: Simple reflex agents use **condition-action rules** to determine their actions. For example, in a vacuum cleaner environment, a rule might be:
    - If the current location is dirty, then suck the dirt.
  - **Decision Making**: The agent evaluates its current percept, checks it against its predefined rules, and performs the corresponding action. For instance, if the agent sees that the location is dirty, it cleans. If the location is clean, it doesn't take action based on that percept alone.
- **Examples**
- **Vacuum Cleaner**: A simple reflex vacuum agent might have rules like:
  - If [Dirty], then Suck.
  - If [Clean], then Move Left or Right randomly.

Limitations

- Limited Intelligence :
  - Simple reflex agents are not very intelligent because they only respond to the current percept and cannot handle complex situations or sequences of actions. They work well in fully observable environments where all necessary information is visible in the current percept.
- Handling Partial Observability:
  - If the environment is not fully observable or if the agent lacks complete information, simple reflex agents can struggle. For example, if a vacuum agent can only detect dirt but not its location, it might end up in an infinite loop, moving back and forth without completing its task.
- Randomization :
  - To avoid some of the issues like infinite loops, simple reflex agents can use randomization. For example, if the agent doesn't know which direction to move in a clean environment, it might randomly choose between moving left or right.

Model-based reflex agents

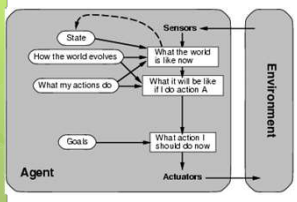



```
function MODEL-BASED-REFLEX-AGENT(percept) returns an action
  persistent: state, the agent's current conception of the world state
              model, a description of how the next state depends on current state and action
              rules, a set of condition–action rules
              action, 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.

- An agent must maintain an internal state to track aspects of the world it cannot observe directly at any given moment. This internal state is updated based on percept history
- **Knowledge for Updating State**:
  - **World Evolution**: The agent needs to know how the world evolves independently of its actions. For instance, an overtaking car gets closer over time.
  - **Effects of Actions**: The agent must understand how its actions affect the world. For example, turning the steering wheel clockwise makes the car turn right.
- The agent updates its internal state by combining the current percept with the old internal state, based on its model of the world.
- Uncertainty and Decision-Making: Due to partial observability, an agent often cannot determine the exact current state. Instead, it operates on its best guess of the state and still needs to make decisions under uncertainty.

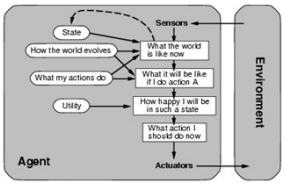
Goal-based agents



•knowing state and environment? Enough?
–Taxi can go left, right, straight
•Have a goal
A destination to get to Uses knowledge about a goal to guide its actions
E.g., Search, planning

- Knowing the **current state isn't always enough to decide what to do**. For instance, at a road junction, a taxi's decision to turn left, right, or go straight depends on its destination.
- Goal-based agents need information **about desirable outcomes**, such as reaching the passenger's destination.
- The agent combines **goal information with a model** of how actions affect the world (similar to the model-based reflex agent) to choose actions that achieve the goal.
- goal-based agents consider future outcomes. They ask questions like **"What will happen if I do this?"**
- Goal-based agents are **more flexible** because they can update their knowledge and goals easily. For example, they can adjust behavior based on new conditions like rain or change their destination by updating the goal.
  - While reflex agents might appear more efficient due to their straightforward condition-action rules, they are less adaptable. Changing conditions or goals would require rewriting many rules.
- In **complex situations**, achieving a goal may involve considering long sequences of actions. **Search and planning** are AI subfields dedicated to finding these action sequences.

Utility-based agents



Goals are not always enough
- Many action sequences get taxi to destination
- Consider other things. How fast, how safe…..

A utility function maps a state onto a real number which describes the associated degree of "happiness", "goodness", "success".

Goals Alone Are Insufficient

- Goals provide a binary distinction between "happy" (goal achieved) and "unhappy" (goal not achieved) states, but they don't distinguish between the quality of achieving the goal.
- **In a taxi service,** many routes can get a passenger to their destination, but some are quicker, safer, more reliable, or cheaper.

Utility as a General Performance Measure

- Utility allows comparison of different world states based on how desirable they are, providing a more nuanced measure than simple goals.
- **In the taxi service**, utility can evaluate routes based on factors like time, cost, and safety.

Utility Function and Rationality

- An agent's utility function internalizes the performance measure. If aligned with the external performance measure, maximizing utility equates to rational behavior.
- A **utility-based taxi agent** chooses a route that maximizes a combined measure of time, cost, and safety, rather than just aiming to reach the destination.

Advantages of Utility-Based Agents

- **Flexibility and Learning:** Utility-based agents adapt better and learn from experience.
- **Handling Conflicting Goals:** Utility functions help in making trade-offs between conflicting goals, such as speed and safety.
- **Decision Making Under Uncertainty:** Utility provides a framework to weigh the likelihood of success against goal importance.
- **Example:** A utility-based agent in online shopping weighs cost, delivery time, and customer ratings to choose the best product.

Decision Making Under Uncertainty

- Rational utility-based agents choose actions that maximize expected utility, considering probabilities and utilities of outcomes.
- In uncertain environments, a utility-based taxi agent considers potential traffic delays and the probabilities of different routes being faster.
- Building utility-based agents involves modeling and tracking the environment, requiring advanced perception, representation, reasoning, and learning techniques.

Taxi Service Example

- **Goal-Based Agent:** Any route reaching the destination is acceptable.
- **Utility-Based Agent:** Evaluates routes based on time, cost, and safety. Chooses the route with the highest utility score.

Online Shopping Example

- **Goal-Based Agent:** Completes the purchase regardless of delivery speed, cost, or customer reviews.
- **Utility-Based Agent:** Uses a utility function to weigh item cost, delivery time, shipping cost, and customer ratings. Chooses the option with the highest utility score.

Example 1: Taxi Service

- **Goal-Based Agent:** A goal-based agent for a taxi service has the simple objective of getting passengers to their destinations. However, it does not consider factors like time, cost, or safety. For instance:
- **Route A:** Takes 30 minutes, costs $20, and is very safe.
- **Route B:** Takes 25 minutes, costs $25, and is less safe.
- **Route C:** Takes 40 minutes, costs $15, and is unsafe.
- All routes achieve the goal of reaching the destination, but a goal-based agent doesn't differentiate between them.
- **Utility-Based Agent :** A utility-based agent, on the other hand, evaluates these factors using a utility function:
- **Utility Function:** $U = -(\text{time} \times 2) - \text{cost} - (\text{safety risk} \times 5)$
- **Route A:** $U = -(30 \times 2) - 20 - (1 \times 5) = -60 - 20 - 5 = -85$
- **Route B:** $U = -(25 \times 2) - 25 - (3 \times 5) = -50 - 25 - 15 = -90$
- **Route C:** $U = -(40 \times 2) - 15 - (5 \times 5) = -80 - 15 - 25 = -120$
- The utility-based agent chooses Route A because it has the highest (least negative) utility, balancing time, cost, and safety.

Example 2: Online Shopping

- **Goal-Based Agent:** A goal-based agent for an online shopping platform might simply aim to complete a purchase. It doesn't consider factors like delivery speed, shipping cost, or customer reviews. For example:
- **Option 1:** Item costs $50, 5-day delivery, free shipping, 4-star rating.
- **Option 2:** Item costs $55, 2-day delivery, $5 shipping, 5-star rating.
- **Option 3:** Item costs $45, 7-day delivery, $10 shipping, 3-star rating.
- All options complete the purchase goal, but the agent doesn't differentiate between them.
- **Utility-Based Agent:** A utility-based agent uses a utility function to evaluate options:
- **Utility Function:** $U = -\text{cost} - (\text{delivery time} \times 2) + (\text{rating} \times 10) - \text{shipping cost}$
- Applying this function:
- **Option 1:** $U = -50 - (5 \times 2) + (4 \times 10) - 0 = -50 - 10 + 40 = -20$
- **Option 2:** $U = -55 - (2 \times 2) + (5 \times 10) - 5 = -55 - 4 + 50 - 5 = -14$
- **Option 3:** $U = -45 - (7 \times 2) + (3 \times 10) - 10 = -45 - 14 + 30 - 10 = -39$
- The utility-based agent chooses Option 2 because it has the highest utility, considering cost, delivery time, rating, and shipping cost.