What is AI?

Science of AI

Physics: Where did the physical universe come from?

And what law guide its dynamics?

Biology: How did biological life evolve?

And how do living organisms function?

Artificial Intelligence: What is the nature of intelligent thought?

What is Intelligence?

Dictionary.com: capacity for learning, reasoning, understanding, and similar forms of mental activity

Ability to perceive and act in the world

Reasoning: proving theorems and medical diagnosis

Planning: Take decisions

Learning and adaptation: recommend movies, learn traffic patterns

Understanding: Text, speech, visual scenes

Intelligence vs. Humans

Are humans intelligent?

- Replication human behaviour early hallmark of intelligence

Are humans always intelligent?

Can non-human behaviour be intelligent?

Some Definitions of AI

“ We call programs intelligent if they exhibit behaviors that would be regarded intelligent if they were exhibited by human beings. “

— Herbert Simon

“ Physicists ask what kind of place this universe is and seek to characterize its behavior systematically. Biologists ask what it means for a physical system to be living. We in AI wonder what kind of information-processing system can ask such questions. “

— Avron Barr and Edward Feigenbaum

Some Definitions of AI

“ AI is the study of techniques for solving exponentially hard problems in polynomial time by exploiting knowledge about the problem domain. “

— Elaine Rich

“ AI is the study of mental faculties through the use of computational models. “

— Eugene Charniak and Drew McDermott

What is Artificial Intelligence?

	human-like	rational
thought	“[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 & McDermott 1985)
behavior	“The study of how to make computers do things at which, at the moment, people are better” (Rich & Knight 1991)	“The branch of computer science that is concerned with the automation of intelligent behavior” (Luger & Stubblefield 1993)

human-like vs. rational

thought vs. behavior

Thinking Humanly

- Cognitive Science
  - Very hard to understand how humans think
    - Post-facto rationalizations, irrationality of human thinking
- Do we want a machine that beats humans in chess or a machine that thinks like humans while beating humans in chess?
  - Deep Blue supposedly DOESN'T think like humans
- Thinking like humans important in Cognitive Science applications
  - Intelligent tutoring
  - Expressing emotions in interface.. HCI

Acting Rationally

- Rational behavior: doing the right thing
- Need not always be deliberative
  - Reflexive
- Aristotle (Nicomachean ethics)
  - Every art and every inquiry, and similarly every action and every pursuit is thought to aim at some good.

Acting ⇒ Thinking?

- Weak AI hypothesis vs. Strong AI hypothesis
  - Weak Hyp: machines could act as if they are intelligent
  - Strong Hyp: machines that act intelligent have to think intelligently too

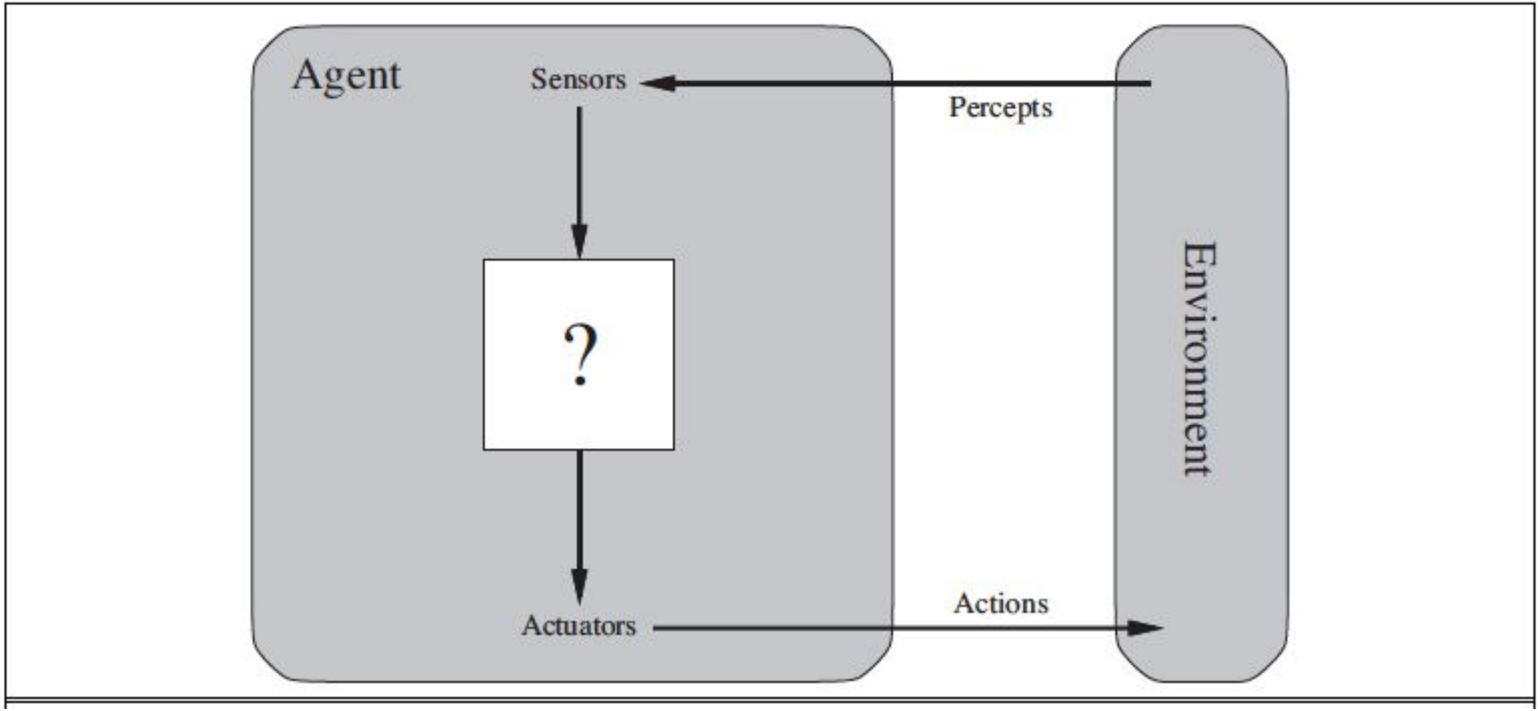
Intelligent Agents

Agent

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

- We use 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.
- “In general, 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.



Agents interact with environments through sensors and actuators.

PEAS (Performance, Environment, Actuator and Sensor) description

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

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

Examples of agent types and their PEAS descriptions.

Rational Agent

A rational agent is one that does the right thing. Obviously, doing the right thing is better than doing the wrong thing, but what does it mean to do the right thing?

Moral philosophy has developed several different notions of the “right thing,” but AI has generally stuck to one notion called consequentialism: we evaluate an agent’s behavior by its consequences.

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

Rationality

What is rational at any given time depends on four things:

- The performance measure that defines the criterion of success.
- The agent's prior knowledge of the environment.
- The actions that the agent can perform.
- The agent's percept sequence to date.

This leads to a definition of a 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."

What is an Intelligent Agent?

An Intelligent Agent (IA) is

- a system that perceives its environment and takes actions to maximize its chances of success.
- These agents can be software-based, hardware-based, or a combination of both,
- and they operate autonomously or semi-autonomously to achieve specific goals.

Characteristics of Intelligent Agents

- **Autonomy:** Agents operate without direct human intervention and have control over their actions and internal state.
  - Self-Driving Cars
- **Perception:** They perceive their environment through sensors or data inputs.
  - Security Surveillance Systems
- **Action:** Agents take actions that affect their environment.
  - Robotic Vacuum Cleaners

Characteristics of Intelligent Agents

- **Rationality:** They act rationally to achieve their goals, which means they make decisions that maximize their expected utility.
    - Automated Trading Systems
- **Learning:** Some agents can improve their performance over time through learning from experiences.
    - Recommendation Systems

Agent Environment

- Definition: The environment in which agents operate.
- Examples:
  - Real-world environments, virtual environments, simulated environments.
- Types of Environments:
  - Fully Observable vs. Partially Observable
  - Deterministic vs. Stochastic
  - Episodic vs. Sequential
  - Static vs. Dynamic
  - Discrete vs. Continuous
  - Single-agent vs. Multi-agent
  - Known vs. Unknown

Fully Observable vs. Partially Observable

Fully Observable Environment:

- The agent has access to the complete state of the environment at each point in time.
- There is no hidden information, allowing the agent to make fully informed decisions.

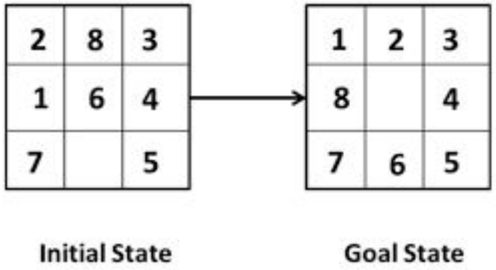
Partially Observable Environment:

- The agent only has partial information about the state of the environment due to limitations in sensors or inherent characteristics of the environment.

Deterministic vs. Stochastic

Deterministic Environment:

- The next state of the environment is completely determined by the current state and the actions performed by the agent.
- There is no uncertainty in the outcome of actions.



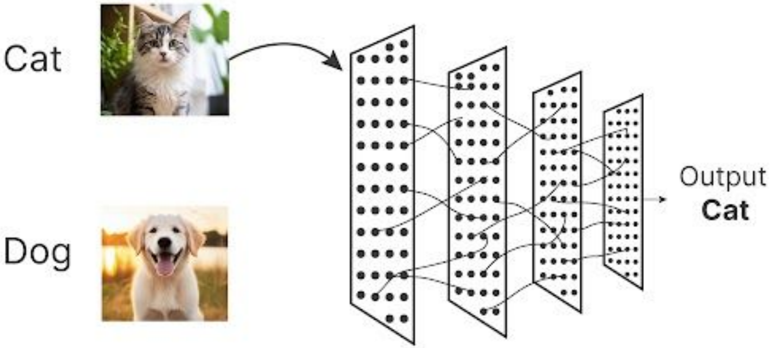
Stochastic Environment:

- The next state of the environment is not fully predictable due to random elements or inherent uncertainty.
- Actions may lead to different outcomes with certain probabilities.

Episodic vs. Sequential

Episodic Environment:

- The agent's actions are divided into discrete episodes.
- Each episode is independent of the others, and the outcome of one episode does not affect the next.



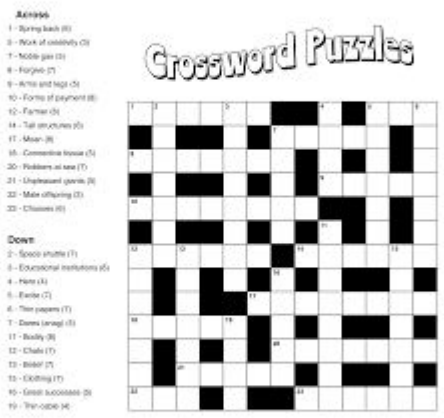
Sequential Environment:

- The current decision could affect all future decisions.
- The agent's actions are part of a continuous sequence, and each action influences subsequent actions.

Static vs. Dynamic

Static Environment:

- The environment remains unchanged while the agent is making a decision.
- The agent does not need to worry about the environment changing in the middle of its decision process.



Dynamic Environment:

- The environment can change while the agent is making a decision, requiring the agent to adapt to changes in real-time.

Discrete vs. Continuous

Discrete Environment:

- The environment consists of a finite number of distinct states and actions.
- The agent’s decisions are based on a limited set of possibilities.

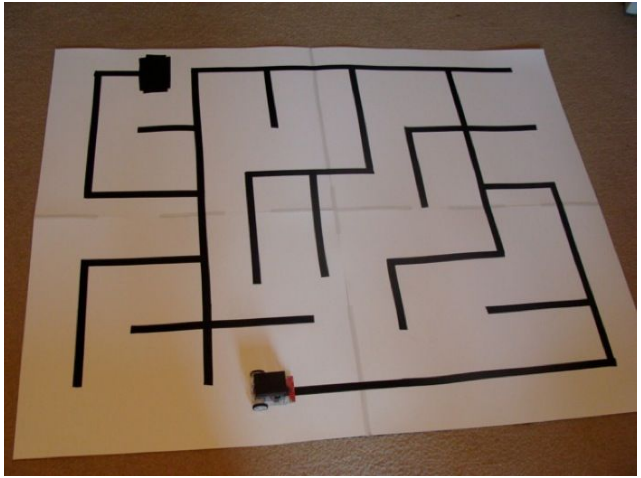
Continuous Environment:

- The environment has a continuous range of states and actions.
- The agent must handle a potentially infinite number of possibilities.

Single-agent vs. Multi-agent

Single-agent Environment:

- Only one agent is operating in the environment,
- and there are no other agents to interact with or compete against.



Multi-agent Environment:

- Multiple agents operate within the environment, interacting with each other.
- These interactions can be competitive or cooperative.

Known vs. Unknown

Known Environment:

- The agent has complete knowledge of the environment’s dynamics, including how actions affect the state of the environment.
- This allows for precise planning and decision-making.

Unknown Environment:

- The agent has no prior knowledge of the environment's dynamics
- and must learn how actions affect states through exploration and interaction.

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

Examples of task environments and their characteristics

Types of Agents

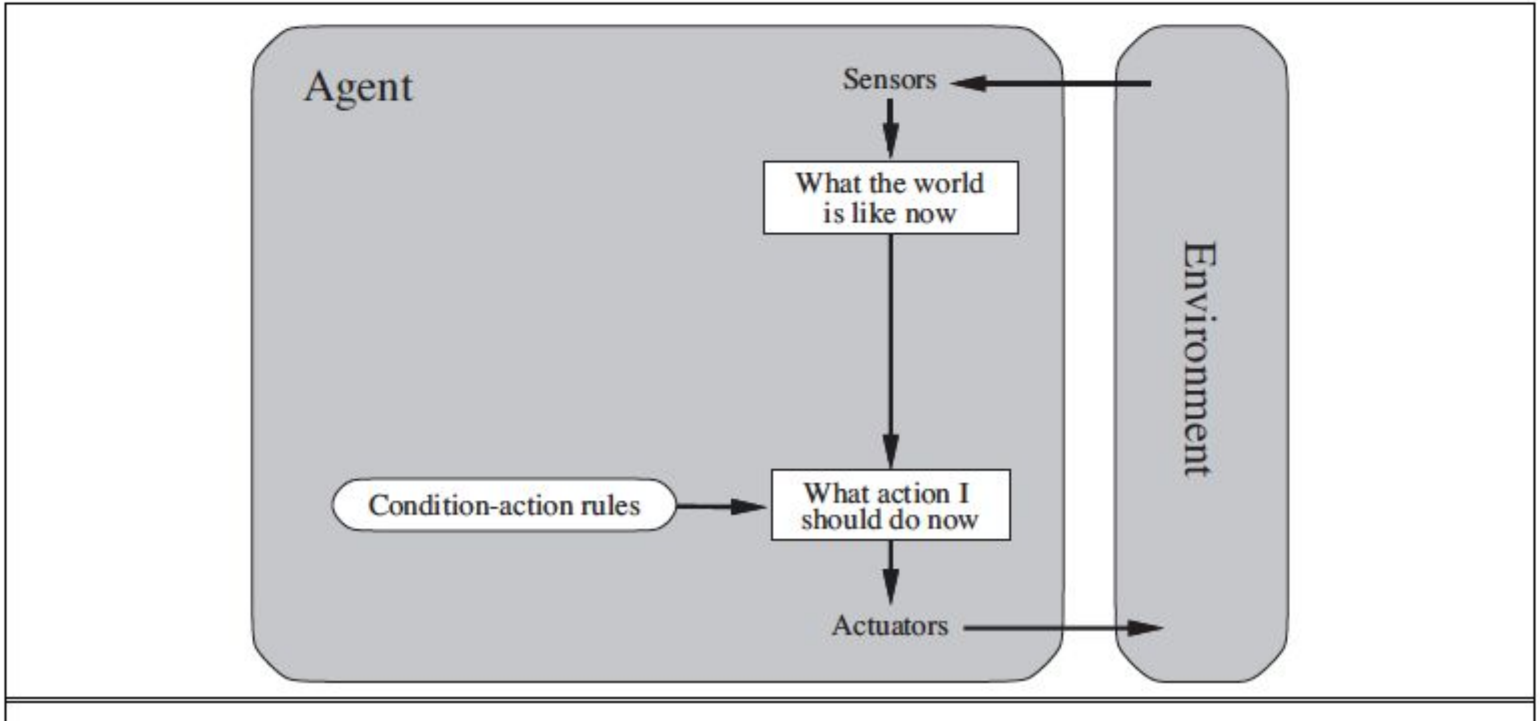
- Simple Reflex Agents
- Model-Based Reflex Agents
- Goal-Based Agents
- Utility-Based Agents

Simple Reflex Agents (SRA)

Simple Reflex Agents operate by selecting actions based solely on the current percept (i.e., what they sense at the moment), ignoring the rest of the percept history. They follow condition-action rules (if-then statements) to make decisions. They do not consider the history of percepts or the internal state of the environment.

How Simple Reflex Agents Work

- Perception: The agent perceives the environment through its sensors.
- Condition-Action Rules: The agent evaluates the current percept against a set of predefined rules.
- Action: Based on the matching condition, the agent performs the corresponding action.



Schematic diagram of a simple reflex agent.

SRA: Characteristics

1. **Direct Response**:
    - Simple reflex agents respond directly to percepts with actions. They do not maintain any internal state or history of previous percepts.
2. **Condition-Action Rules**:
    - The behavior of these agents is governed by a set of rules, often referred to as "if-then" rules. For example, "if the car in front is braking, then decelerate."
3. **No Memory**:
    - These agents do not have memory; they do not keep track of past percepts or actions. Their decisions are made purely on the basis of the current percept.
4. **Limited Applicability**:
    - Simple reflex agents are effective in environments that are fully observable and predictable. However, they are less effective in complex or partially observable environments where decisions need to consider past events or future consequences.

SRA: Strengths and Limitations

Strengths:

- **Simplicity**: Simple reflex agents are easy to design and implement due to their straightforward nature.
- **Efficiency**: In well-defined, predictable environments, they can operate efficiently without the need for complex processing.

Limitations:

- **Lack of Adaptability**: Without memory or consideration of the history, these agents cannot adapt to changes or learn from past experiences.
- **Limited Scope**: They are not suitable for complex environments where the current percept alone does not provide enough information to make an optimal decision.

SRA Example

Example: A basic thermostat:

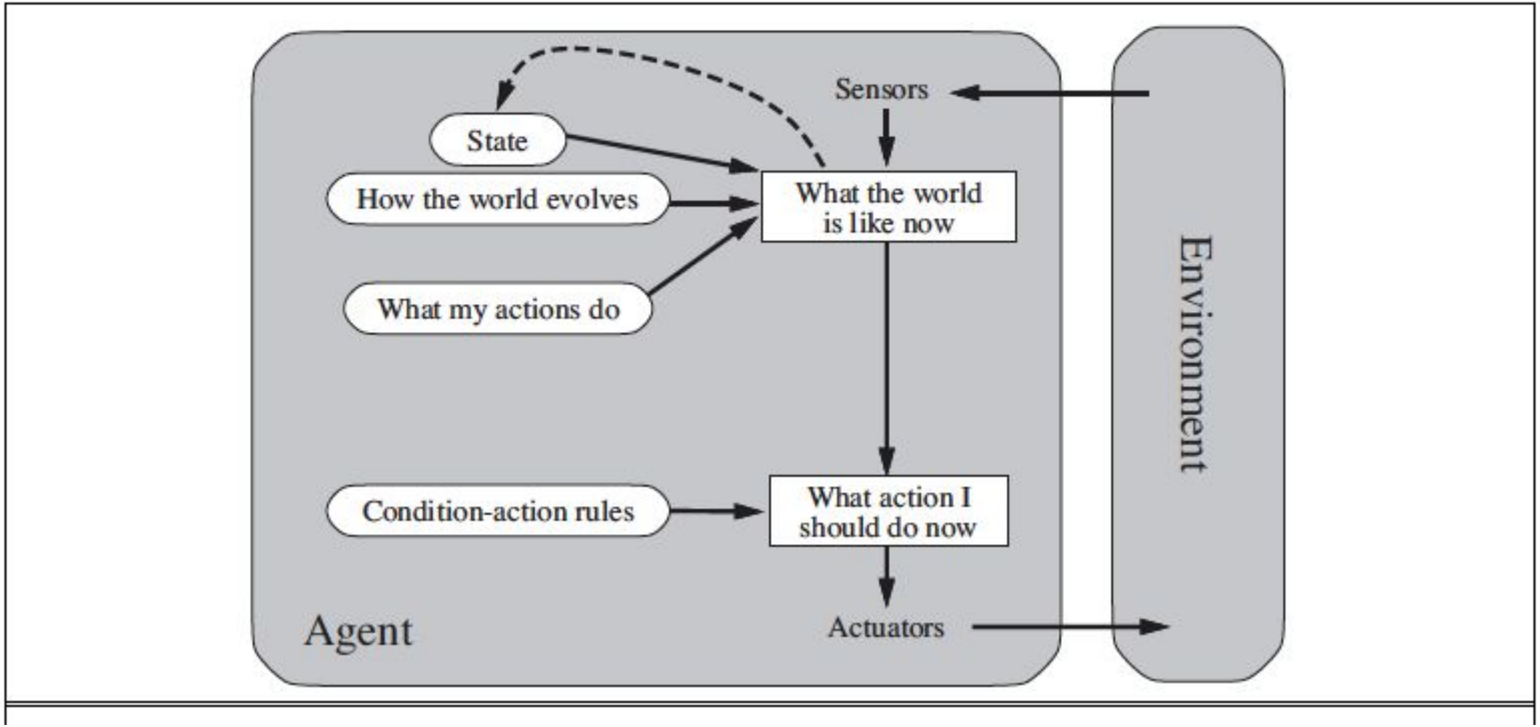
- **Percept**: Current room temperature.
- **Condition-Action Rule**: If the temperature is below 20°C, turn on the heater; if the temperature is above 25°C, turn off the heater.
- **Action**: Turning the heater on or off based on the current temperature.

Model-Based Reflex Agents (MBRA)

Model-Based Reflex Agents maintain an internal state that depends on the percept history and a model of how the world works. This model helps the agent to keep track of parts of the world it can't see at the moment and to make decisions based on a combination of current percepts and stored information.

How Model-Based Reflex Agents Work

1. **Perception**: The agent perceives the environment through its sensors.
2. **Update State**: The agent updates its internal state based on the current percept and its internal model.
3. **Condition-Action Rules**: The agent evaluates its internal state and current percept against a set of predefined rules.
4. **Action**: Based on the matching condition, the agent performs the corresponding action.



Schematic diagram of a model-based reflex agent

MBRA: Characteristics

1. **Internal State**:
   - Model-based reflex agents maintain an internal state that depends on the percept history. This internal state captures relevant aspects of the environment that the agent cannot directly perceive at any given time.
2. **World Model**:
   - These agents use a model of the world that describes how the world evolves in response to actions and how percepts are generated. This model helps the agent update its internal state based on new percepts.
3. **Condition-Action Rules**:
   - Like simple reflex agents, model-based reflex agents use condition-action rules. However, these rules are applied to the internal state rather than directly to the percepts.
4. **State Updating**:
   - The agent updates its internal state using the model of the world whenever it receives a new percept. This process involves incorporating new information into the existing state.

MBRA: Strengths and Limitations

Strengths:

- **Memory**: The ability to maintain an internal state allows these agents to handle more complex and dynamic environments.
- **Increased Applicability**: Model-based reflex agents can function effectively in partially observable environments where not all relevant information is available at once.
- **Improved Decision Making**: By using an internal model, these agents can anticipate the outcomes of their actions better than simple reflex agents.

Limitations:

- **Complexity**: Maintaining and updating an internal state and world model increases the complexity of the agent.
- **Computation**: The need to update the internal state and apply condition-action rules to this state can require more computational resources.

MBRA: Example

Example: A robotic vacuum cleaner:

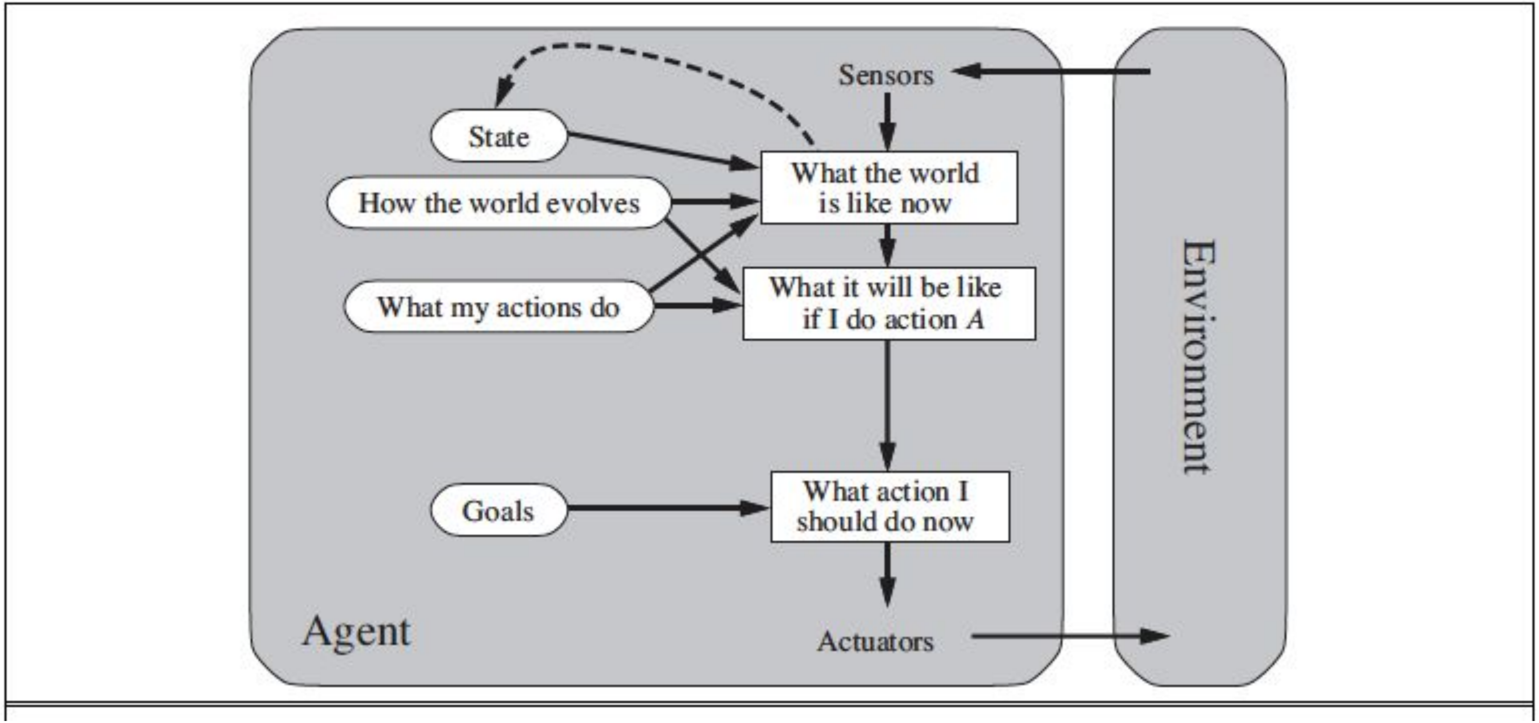
- **Percept**: Current position, obstacles detected.
- **Internal State**: Map of the environment, battery level, cleaning status.
- **Condition-Action Rule**: If the battery level is low and near the charging station, return to the charging station; if an obstacle is detected, change direction.
- **Action**: Moving to the charging station, changing direction, continuing cleaning.

Goal-Based Agents (GBA)

Goal-Based Agents make decisions based on achieving specific goals. These agents evaluate different actions by considering how well each action will help them achieve their goals. They often use planning and search algorithms to find sequences of actions that lead to the desired outcome.

How Goal-Based Agents Work

1. **Perception**: The agent perceives the environment through its sensors.
2. **Goal Identification**: The agent has one or more goals to achieve.
3. **Planning**: The agent evaluates possible actions and sequences of actions to determine which ones will lead to achieving the goal.
4. **Action**: The agent performs the action that is most likely to achieve its goal.



A model-based, goal-based agent. It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

GBA Characteristics

1. **Goals**:
    - Goals are specific desired outcomes or states that the agent aims to achieve. These goals provide a purpose and direction for the agent's actions.
2. **Search and Planning**:
    - Goal-based agents often employ search and planning algorithms to determine a sequence of actions that will lead them from their current state to the goal state. This involves exploring possible future states and evaluating the best path to the goal.
3. **Decision Making**:
    - These agents consider not just the immediate effects of their actions but also their long-term consequences. This forward-thinking approach enables them to handle more complex tasks and environments.
4. **Adaptability**:
    - Goal-based agents can adapt to changes in the environment by re-evaluating their plans and making new decisions based on updated information.

GBA: Strengths and Limitations

Strengths:

- **Effective in Complex Environments**: Goal-based agents can handle complex, dynamic environments where planning and foresight are necessary.
- **Purpose-Driven**: The use of goals provides clear criteria for success, enabling agents to focus their efforts effectively.
- **Flexible and Adaptive**: These agents can re-plan and adapt to changes in the environment, maintaining their focus on achieving their goals.

Limitations:

- **Computational Complexity**: Planning and search algorithms can be computationally intensive, especially in large or complex environments.
- **Requires Clear Goals**: The effectiveness of a goal-based agent depends on well-defined, achievable goals. Ambiguous or conflicting goals can hinder performance.

GBA Example

Example: A GPS Navigation System

- **Perception**: Current location, traffic data, map data.
- **Goal Identification**: Destination address.
- **Planning**: Calculating the optimal route considering current traffic conditions.
- **Action**: Providing turn-by-turn directions to follow the optimal route.

Use Cases:

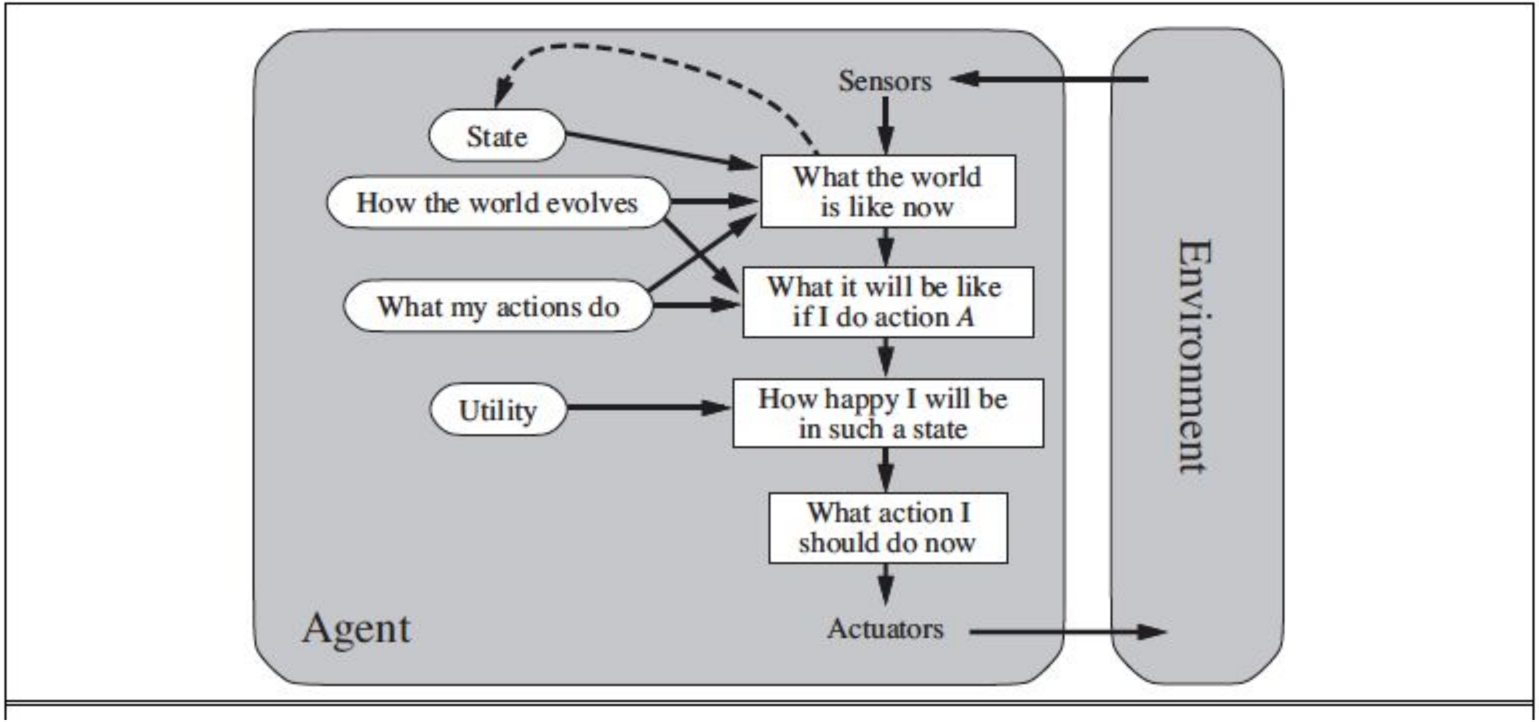
- **Robotics**: A warehouse robot tasked with retrieving an item and delivering it to a specific location.
- **Game AI**: Characters in video games that need to achieve specific objectives, such as capturing a flag or completing a mission.
- **Autonomous Vehicles**: Cars that plan routes to reach destinations efficiently.

Utility-Based Agents (UBA)

Utility-Based Agents extend goal-based agents by considering not only whether goals are achieved but also the desirability (utility) of different states. They use a utility function to evaluate the desirability of different states or outcomes and choose actions that maximize their expected utility.

How Utility-Based Agents Work

1. **Perception**: The agent perceives the environment through its sensors.
2. **Utility Calculation**: The agent uses a utility function to evaluate the desirability of different states or outcomes.
3. **Decision Making**: The agent evaluates possible actions and chooses the one that maximizes expected utility.
4. **Action**: The agent performs the action that maximizes its utility.



A model-based, utility-based agent. It uses a model of the world, along with a utility function that measures its preferences among states of the world. Then it chooses the action that leads to the best expected utility, where expected utility is computed by averaging over all possible outcome states, weighted by the probability of the outcome.

UBA: Characteristics

1. **Utility Function**:
    - A utility function is a mathematical representation that assigns a numerical value (utility) to each possible outcome. This value reflects the agent's preference for that outcome.
2. **Decision Making**:
    - Utility-based agents use the utility function to evaluate and compare different outcomes, choosing the action that maximizes the expected utility. This involves considering not only the likelihood of different outcomes but also their desirability.
3. **Handling Trade-offs**:
    - These agents can handle trade-offs between conflicting goals or objectives. For example, an agent might balance the trade-off between speed and safety in a navigation task.
4. **Risk Management**:
    - Utility-based agents can manage uncertainty and risk by considering the expected utility of actions, which accounts for both the probability and the utility of different outcomes.

UBA: Strengths and Limitations

Strengths:

- **Optimal Decision Making**: Utility-based agents aim to make optimal decisions by maximizing the overall utility, leading to better outcomes.
- **Flexibility**: These agents can adapt to different situations and trade-offs by adjusting their utility function.
- **Handling Complex Environments**: Utility-based agents are well-suited for complex, dynamic environments where multiple factors need to be considered.

Limitations:

- **Complexity**: Defining and computing the utility function can be complex, especially in environments with many variables and uncertainties.
- **Subjectivity**: The utility function is subjective and must be carefully designed to reflect the agent's preferences accurately.

UBA: Example

Example: An Automated Trading System

- **Perception**: Market data, historical trends, economic indicators.
- **Utility Calculation**: Evaluating potential trades based on expected return and risk.
- **Decision Making**: Choosing trades that maximize expected utility (profit adjusted for risk).
- **Action**: Executing buy or sell orders.

Use Cases:

- **Healthcare**: Systems that recommend treatments by balancing efficacy, side effects, and patient preferences.
- **Finance**: Investment algorithms that choose portfolios by balancing expected returns and risks.
- **Resource Management**: Systems that allocate resources (like compute power in a data center) to maximize overall efficiency and performance.

Learning Agents

Learning Agents are capable of improving their performance over time by learning from their experiences. They adapt their behavior based on past outcomes, feedback, and interactions with the environment. Learning agents are designed to handle dynamic and complex environments where predefined rules might not be sufficient.

Key Components of Learning Agents

1. **Learning Element**: Responsible for making improvements and acquiring new knowledge or skills based on experience.
2. **Performance Element**: Executes actions based on the knowledge it has acquired. This component is similar to the decision-making part of non-learning agents.
3. **Critic**: Provides feedback to the learning element about the agent's performance. It evaluates the actions taken and helps the learning element to make adjustments.
4. **Problem Generator**: Suggests actions that lead to new experiences and opportunities for learning. It helps the agent explore new possibilities and gather diverse experiences.

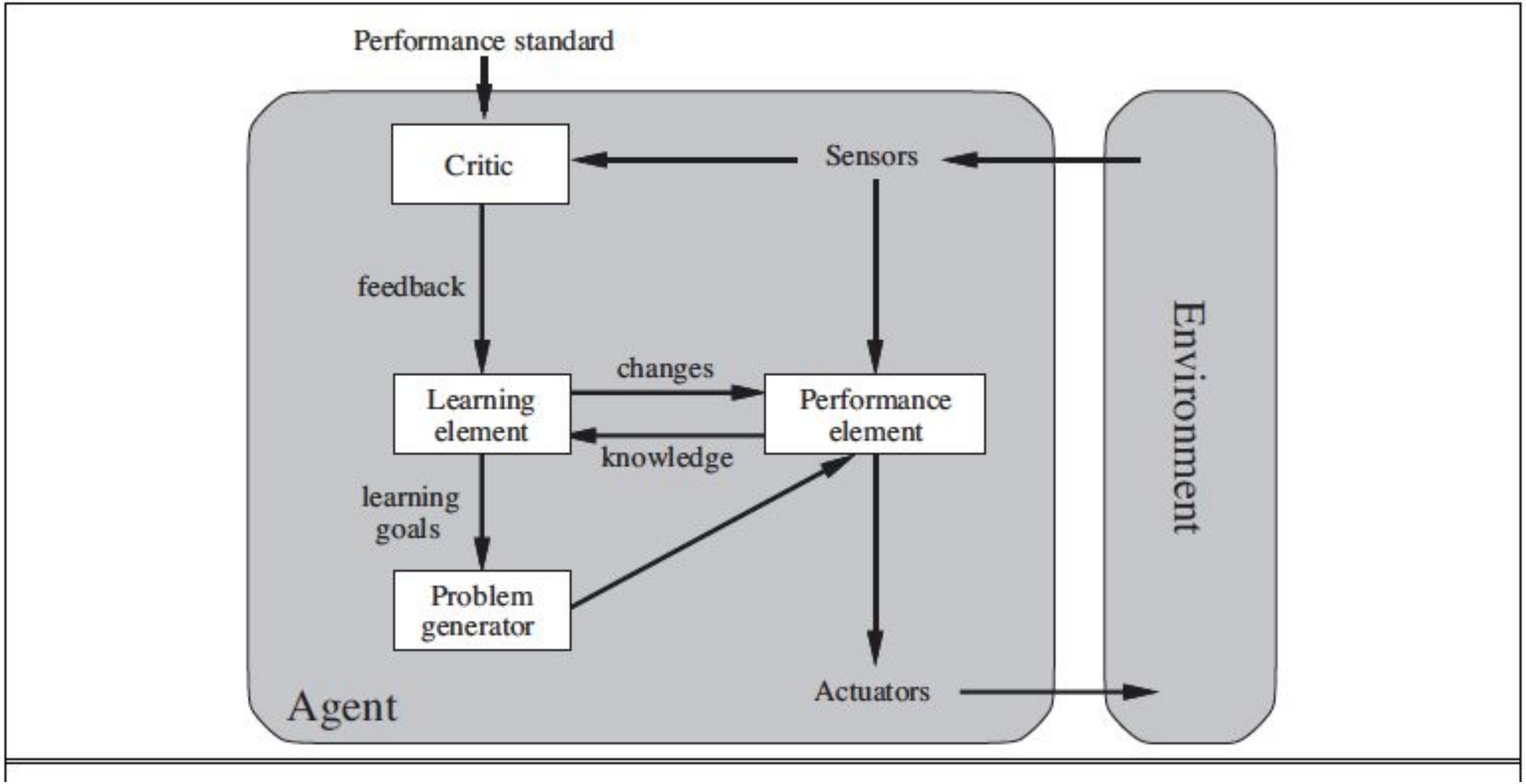
Learning Agents

How Components Contribute to Learning

1. **Learning Element**:
   - **Function**: Updates the agent's knowledge base or policies based on feedback and new data.
   - **Contribution**: Enables the agent to adapt and improve by refining its understanding of the environment and the consequences of its actions.
2. **Performance Element**:
   - **Function**: Uses the current knowledge to make decisions and take actions in the environment.
   - **Contribution**: Acts as the operational part of the agent, implementing the learned strategies and behaviors.

Learning Agents

3. **Critic**:
    - **Function**: Evaluates the outcomes of the agent's actions and provides feedback.
    - **Contribution**: Helps the learning element understand the effectiveness of different actions and identify areas for improvement.
4. **Problem Generator**:
    - **Function**: Encourages exploration by suggesting new actions or strategies that the agent has not tried before.
    - **Contribution**: Promotes learning by exposing the agent to new experiences and challenges, preventing it from getting stuck in suboptimal behaviors.



A general learning agent

Types of Learning in Agents:

Supervised Learning:

- In supervised learning, the agent is provided with labeled training data that includes input-output pairs. The agent learns to map inputs to outputs based on this data, refining its model to improve accuracy.
- Example: A learning agent that recognizes handwritten digits might be trained on a dataset of images labeled with the correct digit.

Unsupervised Learning:

- Unsupervised learning involves training the agent on data without explicit labels. The agent must identify patterns, clusters, or structures within the data on its own.
- Example: A learning agent tasked with customer segmentation might analyze purchase data to identify groups of similar customers without predefined categories.

Types of Learning in Agents:

Reinforcement Learning:

- In reinforcement learning, the agent learns by interacting with its environment and receiving feedback in the form of rewards or punishments. The goal is to learn a policy that maximizes cumulative rewards over time.
- Example: A robot navigating a maze might receive a reward for reaching the goal and penalties for hitting walls, learning to optimize its path to the goal.

Online Learning:

- Online learning refers to the process where the agent learns incrementally as new data arrives, updating its model continuously rather than relying on a fixed dataset.
- Example: A recommendation system that adapts in real-time based on users' interactions with recommended content.

Examples of Learning Agents

1. Supervised Learning Agents

Example: Spam Email Classifier

- **Description**: A spam email classifier learns to differentiate between spam and non-spam emails based on labeled training data.
- **Components**:
  - **Learning Element**: Uses labeled examples to update its classification model.
  - **Performance Element**: Classifies incoming emails based on the learned model.
  - **Critic**: Evaluates the classifier's accuracy using feedback from users (e.g., marking emails as spam or not spam).
  - **Problem Generator**: Incorporates new types of emails into the training set to improve classification.

Examples of Learning Agents

2. Unsupervised Learning Agents

Example: Customer Segmentation

- **Description**: A marketing agent that segments customers into different groups based on purchasing behavior without labeled data.
- **Components**:
  - **Learning Element**: Uses clustering algorithms to group customers based on similarities in data.
  - **Performance Element**: Applies the learned segments to target marketing campaigns.
  - **Critic**: Analyzes the success of marketing campaigns to refine the segmentation.
  - **Problem Generator**: Identifies new customer attributes to consider for segmentation.

Examples of Learning Agents

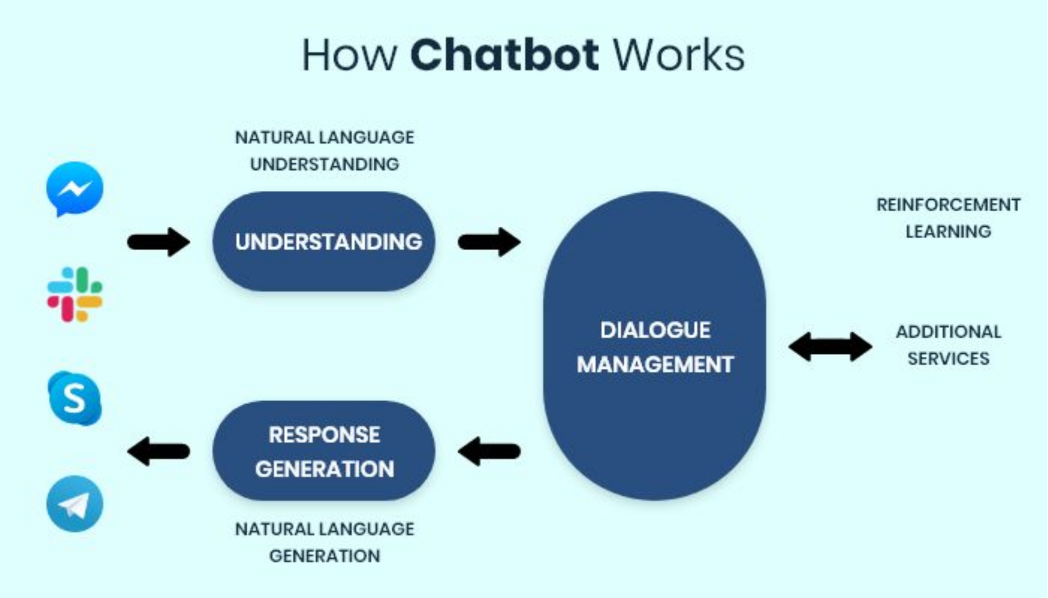
3. Reinforcement Learning Agents

Example: AlphaGo

- **Description**: AlphaGo, developed by DeepMind, uses reinforcement learning to play the game of Go. It learns optimal strategies by playing numerous games against itself and other opponents, constantly improving based on the outcomes.
- **Components**:
    - **Learning Element**: Updates its policy based on the results of games played.
    - **Performance Element**: Executes moves during gameplay.
    - **Critic**: Evaluates the outcome of each game to determine the success of different strategies.
    - **Problem Generator**: Explores new strategies and move sequences to enhance learning.

Case Study: Virtual Personal Assistants

- **Learning Element**: The learning element in a chatbot is responsible for improving the chatbot's ability to understand and generate responses.
- **Performance Element**: The performance element involves the actual operation of the chatbot during interactions with users. It selects responses based on the current state of the conversation
- **Critic**: The critic evaluates how well the chatbot performs in its interactions. (User feedback, sentiment analysis)
- **Problem Generator**: The problem generator in a chatbot might involve testing new types of interactions or exploring different conversation paths to expand the chatbot capabilities.



Other Examples of Intelligent Agents

- **Smart Home Systems**: Systems like Nest smart thermostats and Philips Hue smart lighting adapt to user preferences and environmental conditions. They learn from user behavior to optimize energy usage and provide comfort.
- **Robotics**: Robots in manufacturing, healthcare, and service industries perform tasks autonomously or semi-autonomously. For instance, robotic arms in factories assemble products, and robotic surgery systems assist surgeons in performing precise operations.
- **Game AI**: Non-player characters (NPCs) in video games exhibit intelligent behavior, such as planning, learning, and adapting to player actions. Examples include AI opponents in strategy games like StarCraft or role-playing games like Skyrim.
- **E-commerce Recommendations**: Online retailers like Amazon use intelligent agents to recommend products to users based on their browsing and purchase history. These agents analyze large datasets to identify patterns and suggest items that users are likely to be interested in.

How the components of agent programs work

In agent programs, representations of the environment and internal state can be categorized into atomic, factored, and structured representations. Each type of representation has different implications for how an agent perceives, reasons about, and interacts with the world.

The representations are:

- Atomic
- Factored
- Structured

Atomic Representation

In atomic representations, each state of the environment is treated as a unique, indivisible entity without any internal structure.

Components:

1. **State Representation**:
    - Each state is a distinct and unique identifier.
    - No internal features or attributes are used to describe the state.
2. **Perception**:
    - The agent perceives the environment as a set of distinct states.
    - Each perception corresponds to a specific state.
3. **Decision Making**:
    - The decision-making process involves mapping each state directly to an action.
    - Simple lookup tables or state-action pairs are commonly used.
4. **Action**:
    - Actions are selected based on the current state as perceived.
    - No internal structure within states means actions are typically predefined for each state.

Example:

- **Chess**: Each board configuration is a unique state without any further breakdown of individual pieces or positions.

Factored Representation

In factored representations, states are described by a set of variables (features), each of which can take on different values. This allows for a more detailed and flexible representation of the environment.

Components:

1. **State Representation**:
   - A state is represented as a vector of variables (features).
   - Each variable can take on different values, providing a structured description of the state.
2. **Perception**:
   - The agent perceives the environment by assigning values to the relevant variables.
   - Sensor data is mapped to these variables to create a factored state description.
3. **Decision Making**:
   - Decision-making involves evaluating the values of different variables to determine the best action.
   - Techniques such as decision trees, Bayesian networks, or linear programming can be used.
4. **Action**:
   - Actions are selected based on the values of the state variables.
   - Policies or rules are defined in terms of these variables.

Example:

- **Self-Driving Car**: The state might include variables like the car’s position, speed, nearby obstacles, and traffic signals. Each variable can take on different values, such as specific locations, velocities, or statuses of traffic lights.

Structured Representation

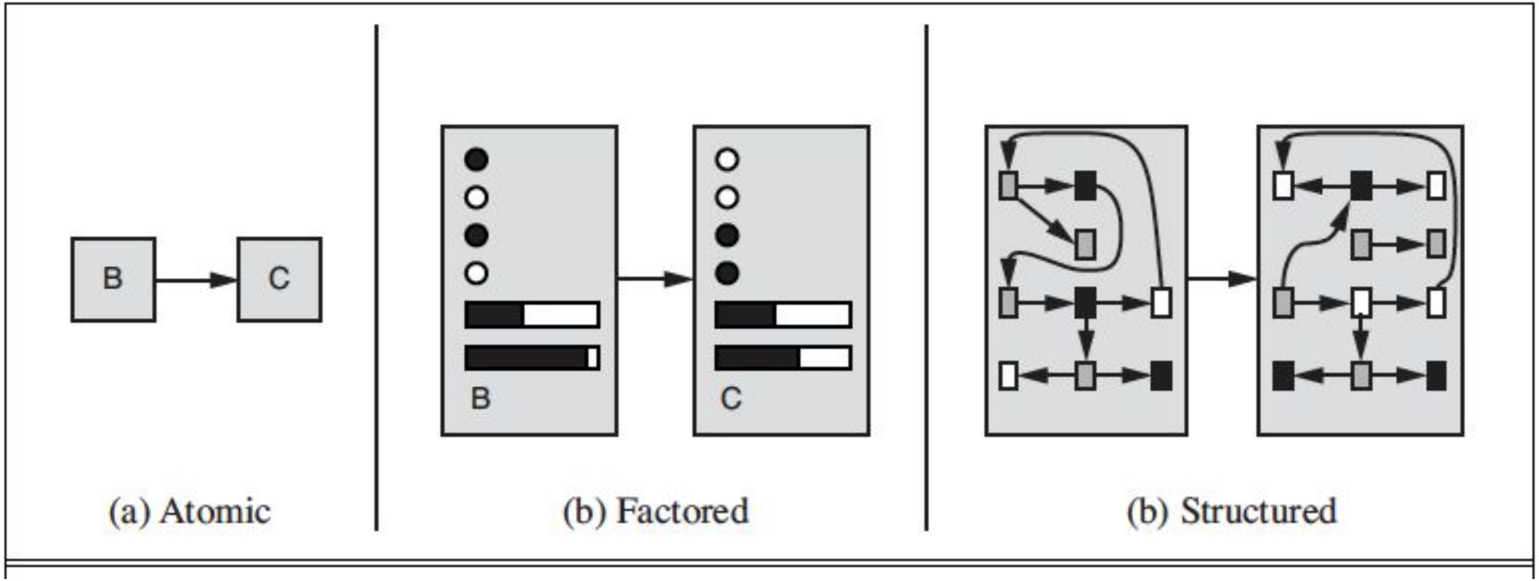
Structured representations go a step further by incorporating relationships between objects and attributes within the state. This allows for complex and rich descriptions of the environment.

Components:

1. **State Representation**:
   - States are represented as a collection of objects, each with its own attributes and relationships to other objects.
   - This is often modeled using graphs, relational databases, or logical representations.
2. **Perception**:
   - The agent perceives the environment by identifying objects, their attributes, and the relationships between them.
   - Advanced perception systems, like those using computer vision or natural language processing, are often required.
3. **Decision Making**:
   - Decision-making involves reasoning about the objects and their interrelationships.
   - Techniques such as knowledge graphs, ontologies, and first-order logic can be used to infer actions.
4. **Action**:
   - Actions are selected based on the detailed structure of the state.
   - Complex planning algorithms like STRIPS or PDDL might be used to generate action sequences.

Example:

- **NLP:** A sentence might be represented using a parse tree, where words are nodes connected by syntactic relationships.



Three ways to represent states and the transitions between them.

Problem Solving by Search

Solving Problems by Searching

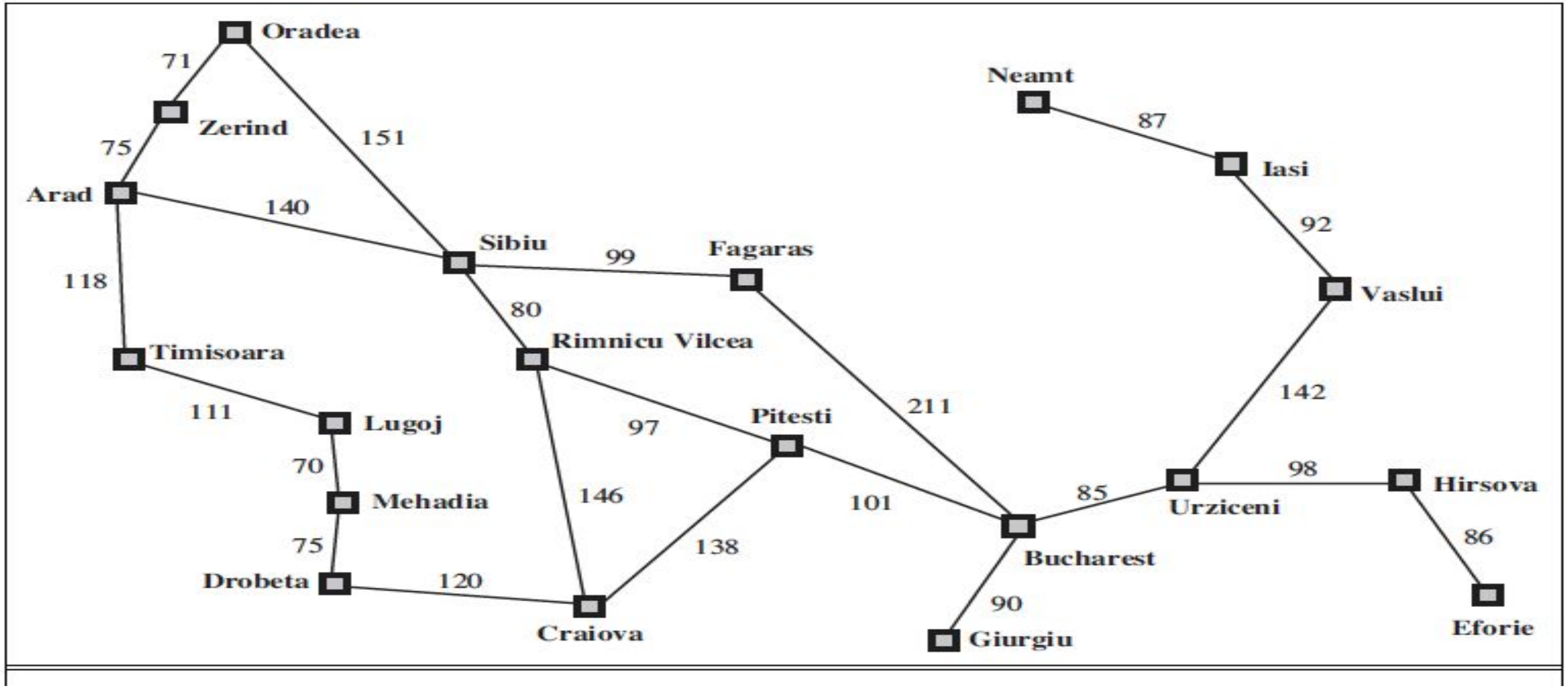
- When the correct action to take is not immediately obvious, an agent may need to plan ahead: to consider a sequence of actions that form a path to a goal state.
- Such an agent is called a problem-solving agent, and the computational process it undertakes is called search.
- Problem-solving agents use atomic representations, that is, states of the world are considered as wholes, with no internal structure visible to the problem-solving algorithms.
- Agents that use factored or structured representations of states are called planning agents.

An Example

- Imagine an agent enjoying a touring vacation in Romania.
- The agent’s performance measure contains many factors: it wants to improve its suntan, improve its Romanian, take in the sights, enjoy the nightlife (such as it is), avoid hangovers, and so on.
- The decision problem is a complex one involving many tradeoffs and careful reading of guidebooks.
- Now suppose the agent is currently in the city of Arad and has a non refundable ticket to fly out of Bucharest the following day.
- In that case, it makes sense for the agent to adopt the goal of getting to Bucharest.
- The agent observes street signs and sees that there are three roads leading out of Arad: one to Sibiu, one to Timișoara, and one to Zerind.

An Example

- None of these are the goal, so unless the agent is familiar with the geography of Romania it will not know which road to follow.
- If the agent has no additional information, i.e., if the environment is unknown then the agent can do no better than to execute one of the actions at random.
- In this class, we will assume our agents always have access to information about the world, such as the map.
- With that information, the agent can follow the following four-phase problem solving process.



A simplified road map of part of Romania

Four-Phase Problem-Solving Process

- Goal Formulation:
    - The agent adopts the goal of getting to Bucharest.
    - Courses of action that don’t reach Bucharest on time can be rejected without further consideration
    - and the agent’s decision problem is greatly simplified.
    - Goals help organize behavior by limiting the objectives that the agent is trying to achieve and hence the actions it needs to consider.
    - Goal formulation, based on the current situation and the agent’s performance measure, is the first step in problem solving.

Four-Phase Problem-Solving Process

- Problem Formulation:
    - We will consider a goal to be a set of world states—exactly those states in which the goal is satisfied.
    - The agent's task is to find out how to act, now and in the future, so that it reaches a goal state.
    - Before it can do this, it needs to decide (or we need to decide on its behalf) what sorts of actions and states it should consider.
    - If it were to consider actions at the level of "move the left foot forward an inch" or "turn the steering wheel one degree left," the agent would probably never find its way out of the parking lot, let alone to Bucharest,
    - because at that level of detail there is too much uncertainty in the world and there would be too many steps in a solution.
    - Problem formulation is the process of deciding what actions and states to consider, given a goal.

Four-Phase Problem-Solving Process

- Search:
  - Before taking any action in the real world, the agent simulate sequences of actions in its model, searching until it find the sequence of actions that reaches the goal.
  - Such a sequence is called a solution.
  - The agent might have to simulate multiple sequences that do not reach the goal, but eventually it will find a solution or it will find that no solution is possible.
  - The process of looking for a sequence of actions that reaches the goal is called search.
  - A search algorithm takes a problem as input and returns a solution in the form of an action sequence.
- Execution:
  - Once a solution is found, the actions it recommends can be carried out.
  - This is called the execution phase.
- Thus, we have a simple "formulate, search, execute" design for the agent.

- It is an important property that in a fully observable, deterministic, known environment *the solution to any problem is a fixed sequence of actions*: i.e, drive to Sibiu, then Fagaras, then Bucharest.
- If the model is correct then once the agent has found the solution it can ignore its percepts while it is executing the action— closing it's eyes, so to speak— because the solution is guaranteed to lead to the goal.
- Control theorist call this an open-loop system: ignoring the percepts break the loop between agent and environment.
- If there is a chance that the model is incorrect or the environment is non deterministic then the agent would be safer using a closed-loop approach that monitors the percepts.

Summary

1. Goal Formulation

Objective: Define what you want to achieve.

2. Problem Formulation

Objective: Translate the goal into a well-defined problem.

3. Search

Objective: Explore the sequence of actions to find a solution path from the initial point to the goal.

4. Execution

Objective: Execute the solution path to achieve the goal.

Distinction Between Well-Defined and Ill-Defined Problems

1. **Clarity and Completeness of Problem Specification:**
   - **Well-Defined:** All elements of the problem (initial state, goal state, operators, constraints) are clearly and completely specified.
   - **Ill-Defined:** One or more elements of the problem are vague, incomplete, or subjective.
2. **Objective vs. Subjective Criteria:**
   - **Well-Defined:** Success criteria are objective and measurable.
   - **Ill-Defined:** Success criteria may be subjective and open to interpretation.
3. **Boundedness of State Space:**
   - **Well-Defined:** The state space is finite and well-delineated.
   - **Ill-Defined:** The state space may be infinite or not clearly defined.

Distinction Between Well-Defined and Ill-Defined Problems

Example Distinction:

- **Well-Defined Problem:** Solving a Sudoku puzzle.
    - **Initial State:** The starting grid with some numbers filled in.
    - **Goal State:** A completed grid that follows the rules of Sudoku.
    - **Operators:** Filling in the blank cells with numbers 1-9 following Sudoku rules.
    - **Constraints:** Each number 1-9 must appear exactly once in each row, column, and 3x3 subgrid.
    - **Criteria for Success:** A fully completed and correct Sudoku grid.
- **Ill-Defined Problem:** Designing a marketing campaign.
    - **Initial State:** General knowledge about the product and target audience.
    - **Goal State:** Increased brand awareness and sales (vague and subjective).
    - **Operators:** Various marketing strategies and tactics (not clearly defined).
    - **Constraints:** Budget limits, market trends (may be ambiguous).
    - **Criteria for Success:** Subjective measures like brand perception, customer engagement, and sales growth.