Intelligent Agents

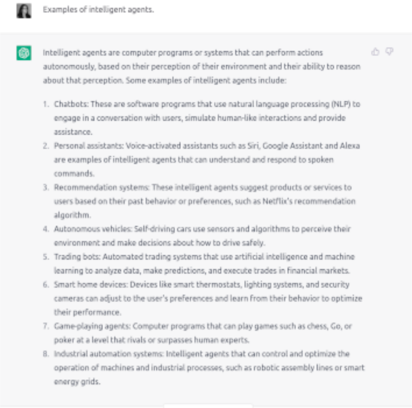
M. Mitrović Dankulov

May 19, 2023

Outline

1. Introduction
2. Environment
3. The structure of agents
4. Agent's organization

Examples of intelligent agents



Examples of intelligent agents.

Intelligent agents are computer programs or systems that can perform actions autonomously, based on their perception of their environment and their ability to reason about that perception. Some examples of intelligent agents include:

1. Chatbots: These are software programs that use natural language processing (NLP) to engage in a conversation with users, simulate human-like interactions and provide assistance.
2. Personal assistants: Voice-activated assistants such as Siri, Google Assistant and Alexa are examples of intelligent agents that can understand and respond to spoken commands.
3. Recommendation systems: These intelligent agents suggest products or services to users based on their past behavior or preferences, such as Netflix's recommendation algorithm.
4. Autonomous vehicles: Self-driving cars use sensors and algorithms to perceive their environment and make decisions about how to drive safely.
5. Trading bots: Automated trading systems that use artificial intelligence and machine learning to analyze data, make predictions, and execute trades in financial markets.
6. Smart home devices: Devices like smart thermostats, lighting systems, and security cameras can adjust to the user's preferences and learn from their behavior to optimize their performance.
7. Game-playing agents: Computer programs that can play games such as chess, Go, or poker at a level that rivals or surpasses human experts.
8. Industrial automation systems: Intelligent agents that can control and optimize the operation of machines and industrial processes, such as robotic assembly lines or smart energy grids.

Rational agents

- Agent - anything that perceives its environment through **sensors** and acting on that environment through **actuators**
- Rational agent - one that behaves as well as possible
- The behavior of agent depends on the environment
- Here we will cover a small set of design principles for building successful agents - intelligent agents

Some basic concepts

- Sensors - means for an agent to receive input from environment; eyes, ears, touch, camera, keystrokes, IR sensors, ...
- Actuators - means for acting on environment; legs, vocal cords, wheels, robotic hands, text, sound, ...
- Percept - agent's perceptual input at any given instant; **percept sequence** is the complete history of everything the agent has ever perceived
- Agent function - maps any given percept sequence to an action

Examples

- Human agent:
  - Sensors: eyes, ears, and other organs.
  - Actuators: hands, legs, mouth, and other body parts.
- Robotic agent:
  - Sensors: Cameras and infrared range finders.
  - Actuators: Various motors.
- Agents everywhere!
  - Thermostat
  - Cell phone
  - Vacuum cleaner
  - Robot
  - Alexa Echo
  - Self-driving car
  - Human

Agent function and program

- Table of all possible percept sequence and agent positions - external characterization of an agent
- Agent function - an abstract mathematical description
- Agent program - a concrete implementation, running within some physical system
- Agent generates a sequence of actions according to the percepts it receives and its agent function

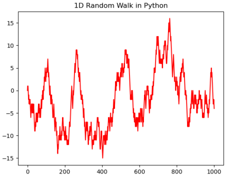
1D random walker

Walker on a 1D grid

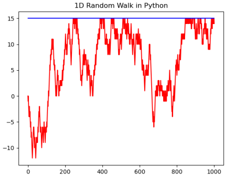
Agent interacts with the wall

Actuator: agent moves one step

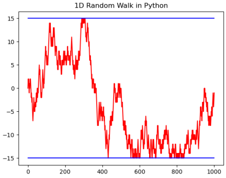
Agent function: at each time step selects to go UP or DOWN with equal probability



No limit



Upper wall



Upper and down wall

Rationality

- Rational agent is one that does the right thing
- Right thing?!
- Performance function measures the success of the agent:
  - it has to be objective
  - not based on wanted agents behavior
  - based on what we want in the environment

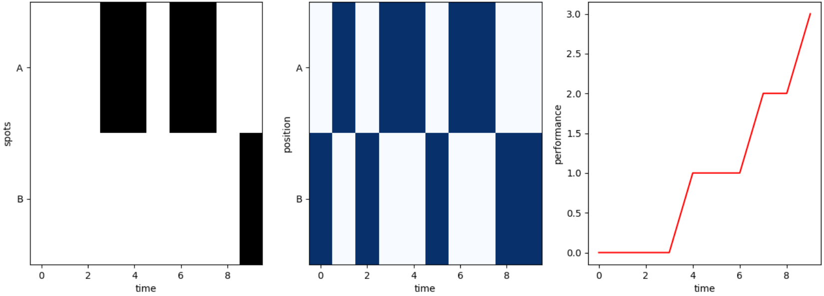
Rational agent

- Factors that influence agent's correctness: performance function, agent's knowledge about the environment, set of actions that agent can perform, percept sequence

- 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
  - Environment: two spots A and B; spots can be clean or dirty
  - Vacuum cleaner actions: left, right and suck
  - Agent's function: if the spot is dirty agent suck, otherwise it moves to other spot

World vacuum cleaner



Rationality is different than perfection!

Rationality maximizes expected performance; perfection maximizes actual performance.

Task environment

Task environment - PEAS - Performance, Environment, Actuators, Sensors

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.

The first step in designing an agent is to define task environment!

Examples of task environment

Agent type	Performance Measure	Environment	Actuators	Sensors
Vacuum cleaner	Cleanness Efficiency Battery life Security	Room Table Wood floor Carpet Various obstacles	Wheels Brushes Vacuum Extractor	Camera Dirt detection sensor Cliff sensor Bump sensor Infrared wall sensor
Medical Diagnose	Healthy patient Minimized cost	Patient Hospital Staff	Tests Treatments	Keyboard (Entry of symptoms)

More examples at:

Properties of task environments I

- Fully vs. partially observable - agents senors; no sensors - environment is unobservable
- Single agent vs. multiagent - crossword puzzle agent - single agent environment; agent playing chess is multiagent environment; multiagent environment - competitive, cooperative, or a combination
- Deterministic vs. stochastic: deterministic - next state of environment completely determined by current state and agent's action; real environments are very complex and thus must be treated as stochastic

Properties of task environments II

- Episodic vs. sequential: episodic - in each episode agent receives a percept and then performs a single action; sequential - current decisions could affect all future decisions;
- Static vs. dynamic: dynamic - environment changes while agent is deliberating; static - environments is not changing while agent decides on its actions
- Discrete vs. continuous - time, percpets and actions determine state of the environment;
- Known vs. unknown: known environment - the outcomes for all actions are given; different from fully and partially observable

Examples

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.

Hardest case - partially observable, multiagent, stochastic, sequential, dynamic, continuous, and unknown

Agent

- AI job - design **agent program** that implements agent function - mapping from percpets to actions
- $agent = agent\ program + architecture$
- Architecture - computing devices with physical sensors and actuators: vacuum cleaner (physical object with wheels, brushes and vacuum extractor), autonomous driving agent - robotic car with different sensors, Chat GPT - a platform with user interface

Agent programs

- Agent program - takes current percept as an input from sensors and transforms it to actions that is then transferred to the actuators
- Agent function - takes the entire percept history; agent program takes the current percept as input

Table Driven Agent

The most simple type of agent

```
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.

Tables are often vast; for realistic tasks are impossible to be written and impossible for agents to learn

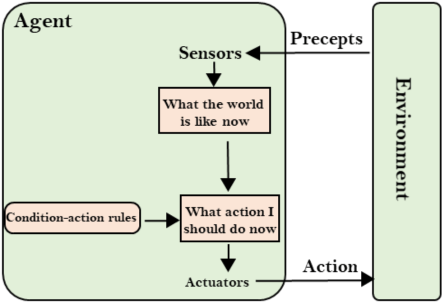
AI task - write the smallest possible agent program that guarantees agent's rationality

Types of agents

- Simple Reflex Agent
- Model-based reflex agent
- Goal-based agents
- Utility-based agent
- Learning agent

Simple Reflex Agent

Decisions are taken on the basis of the current percepts; the rest of the percept history is ignored



Source: https://www.javatpoint.com/

Simple Reflex Agent - properties

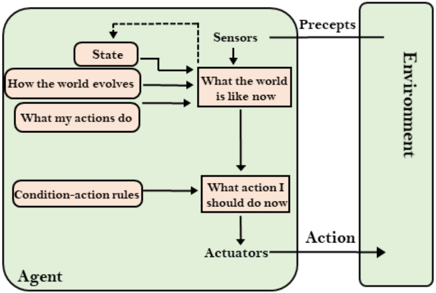
- They exist even in the most complex environment
- They follow condition-action rule - large number of humans actions are based on the same rule (learned or innate reflexes)

Simple Reflex Agent - issues

- They only succeed in the fully observable environment
- Very limited intelligence
- Do not have knowledge of non-perceptual parts of the current state
- Not adaptive to changes in the environment
- Often too big; they can enter infinite loop; can be solved by randomisation

Model-based reflex agent

They can work in a partially observable environment, and track the situation



Source: https://www.javatpoint.com/

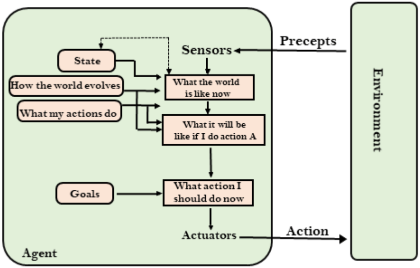
Model-based reflex agent - properties

- Internal state - allow agent to keep track of the part of the world it can?t see now
- Model - how world evolves independent of agent, and how agent's actions influence world
- The Model-Based Reflex Agent can work in a partially observable environment, and track the situation

Goal-based Agents

Knowledge about the current state is not enough

The agent needs to know its goal which describes desirable situations



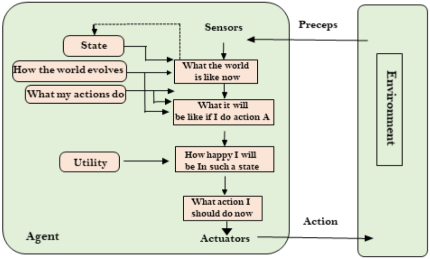
Source: https://www.javatpoint.com/

Goal-based agents

- Goal-based Agents expand the capabilities of the model-based agent by having the "goal" information
- Their choice of action depends on the goal
- These agents may have to consider a long sequence of possible actions before deciding whether the goal is achieved or not; considerations of different scenario are called searching and planning, which makes an agent proactive
- Example chess playing agent

Utility-based agents

Happy and unheppy agent; utility function - measure of success at a given state



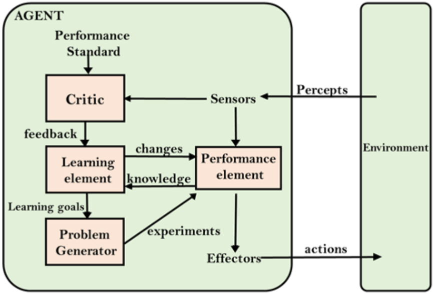
Source: https://www.javatpoint.com/

Utility-based agents - properties

- Their actions depend on goals and best way to achieve them
- They are useful when there are multiple possible alternatives, and an agent has to choose in order to perform the best action
- Utility function maps each state to a real number to check how efficiently each action achieves the goals

Learning agents

Learning agent can learn from its past experiences or it has learning capabilities



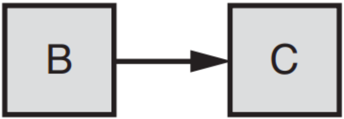
Source: https://www.javatpoint.com/

Learning agents

- Starts with basic knowledge and adapts automatically through learning
- Main components:
  - Learning element - responsible for making improvements by learning from environment
  - Critic - learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard
  - Performance element - responsible for selecting external action
  - Problem generator - responsible for suggesting actions that will lead to new and informative experiences
- Learning agents are able to learn, analyze performance, and look for new ways to improve their performance

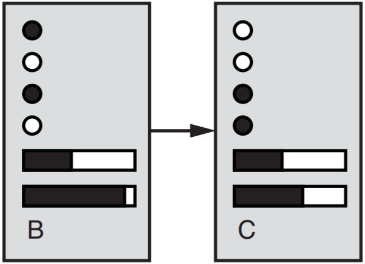
Atomic representation

- Each state of the world is a black box that has no internal structure
- Examples: finding a driving route - each state is a city
- AI algorithms: search, games, Markov decision processes, hidden Markov models, . . .



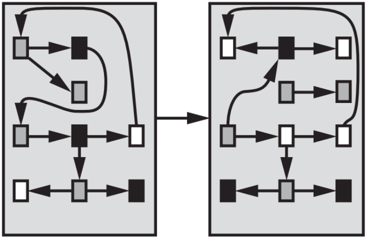
Factored representation

- Each state has some attributevalue properties
- Example: GPS location, amount of gas in the tank
- AI algorithms: constraint satisfaction, and Bayesian networks



Structured representation

- Relationships between the objects of a state can be explicitly expressed
- AI algorithms: first order logic, knowledge-based learning, natural language understanding



Source and credit

Artificial Intelligence, A Modern Approach. Stuart Russell and Peter Norvig. Third Edition. Pearson Education