Topic 2: Intelligent Agents

Tadesse Kebede
taddeekb@gmail.com

Intelligent Agents

- Agents and Environments
- Acting of Intelligent Agents (Rationality)
- Structure of Intelligent Agents
- Agent Types
  - Simple reflex agent
  - Model-based reflex agent
  - Goal-based agent
  - Utility-based agent
  - Learning agent
- How the components of agent programs work

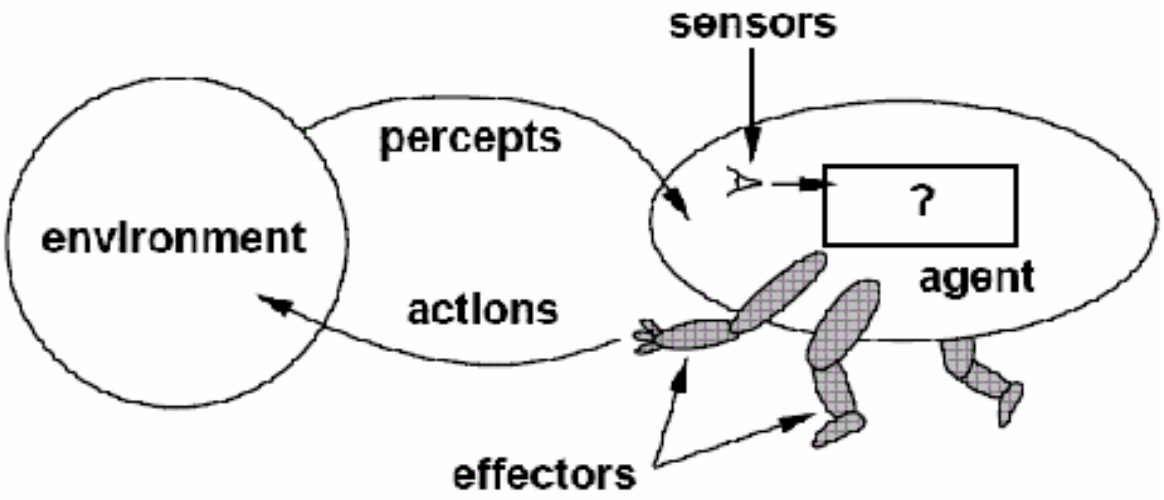
Intelligent Agent

- I want to build a robot that will
  - Clean my house
  - Cook when I don't want to
  - Wash my clothes
  - Cut my hair
  - Fix my car (or take it to be fixed)
  - Take a note when I am in a meeting
  - Handle my emails (Information filtering agent)

  i.e. do the things that I don't feel like doing…
- AI is the science of building software or physical agents that act rationally with respect to a goal.

Agents and Environments

- **Agent** is something that perceives its environment through **SENSORS** and acts upon that environment through **EFFECTORS OR ACTUATORS**.
- The agent is assumed to exist in an environment in which it perceives and acts
- An agent is rational since it does the right thing to achieve the specified goal.



Agent Examples

	Human Agent	Robotic(Physical) Agent	Software agent
Sensors	Eyes, Ears, Nose	Cameras, Scanners, Mic, infrared range finders	receives keystrokes, file contents, and network packets as sensory inputs
Effectors/ Actuators	Hands, Legs, Mouth(Vocal tract)	Various Motors (artificial hand, artificial leg), Speakers, Radio	acts on the environment by displaying on the screen, writing files, and sending network packets

Agents and Environments

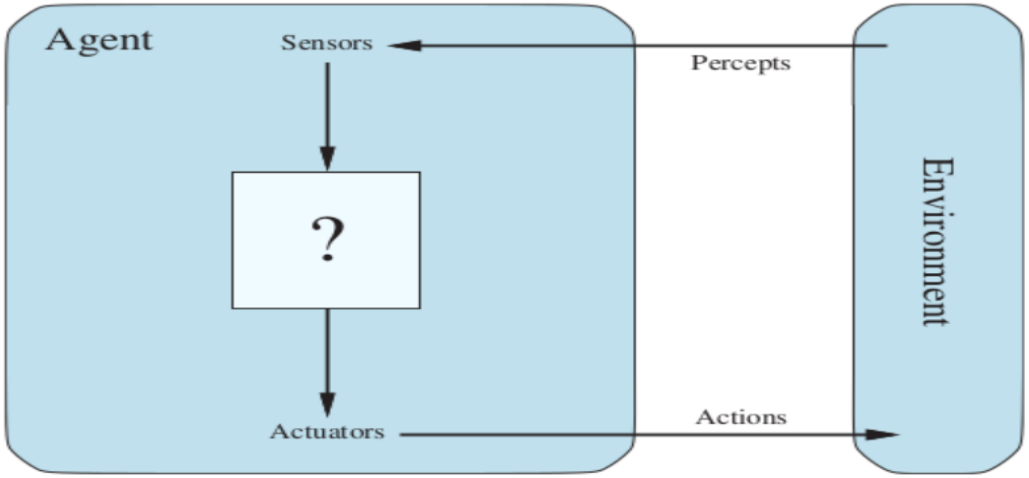


Figure 2.1: Agents interact with environments through sensors and actuators.

- **Percept** refer to the agent's perceptual inputs at any given instant. It is the content an agent's sensors are perceiving.
- 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 its built-in knowledge and on the entire percept sequence observed to date, but not on anything it hasn't perceived.

Agents and Environments

- Mathematically speaking, we say that an agent's behavior is described by the **agent function** that maps any given percept sequence to an action.
- Internally, the agent function for an artificial agent will be implemented by an **agent program**.
- It is important to keep these two ideas distinct.
  - The **agent function** is an abstract mathematical description;
  - the **agent program** is a concrete implementation, running within some physical system.

Illustration of Agents

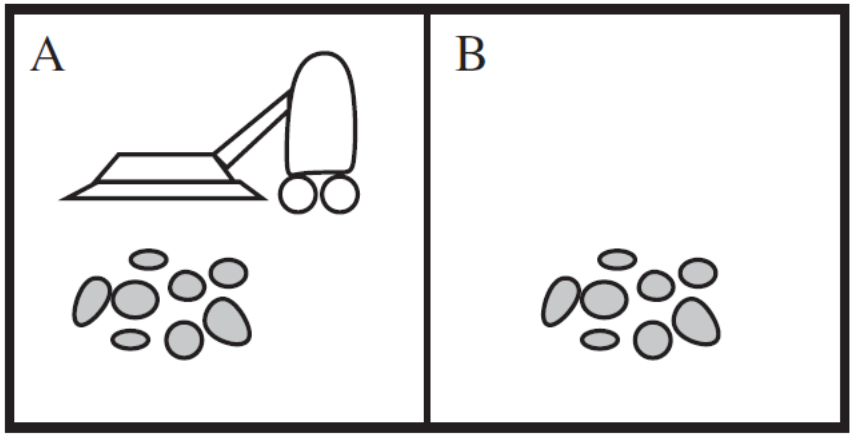


Figure 2.2 A vacuum-cleaner world with just two locations.

- **Sensors Perceive:**
  - This particular world has just two locations: squares A and B.
  - The vacuum agent perceives which square it is in and whether there is dirt in the square.
- **Actions:** It can choose to move left, move right, suck up the dirt, or do nothing.

Illustration of Agents

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

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

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

Illustration of Agents

- An **agent program** that implements a vacuum world with just two locations

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

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

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

Agents and Environments

- An agent is an entity which is:
  - Situated in some environment
  - Autonomous, in the sense that it can act without direct intervention from humans or other software processes, and controls over its own actions and internal state.
  - Flexible which means:
    - Responsive (reactive): agents should perceive their environment and respond to changes that occur in it;
    - Proactive: agents should be able to exhibit opportunistic, goal-directed behavior and take the initiative when appropriate;
    - Social: agents should be able to interact with humans or other artificial agents.

Intelligent Agent

- **An Intelligent Agent**: is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through its effectors to maximize progress towards its goals.
- "Intelligent agents continuously perform **three functions:**
  - Perception of dynamic conditions in the environment;
  - Action to affect conditions in the environment; and
  - Reasoning to interpret perceptions, solve problems, draw inferences, and determine actions."

Types of Intelligent Agents

- Software agents: also called a softbot (software robot)
  - It is an agent that operates within the confines of the computer or computer network.
  - It interacts with a software environment by issuing commands and interpreting the environments feedback.
  - Softbots effectors are commands (e.g. mv or compress in UNIX shell) to change the external environments state.
  - E.g. mail handling agent, information filtering agent
- Physical agents
  - are robots that have the ability to move and act in the physical world and can perceive and manipulate objects in that world, possibly for responding to new perceptions.

Acting of Intelligent Agents (Rationality)

- A rational agent is one that does the right thing—conceptually speaking, every entry in the table for the agent function is filled out correctly. Obviously, doing the right thing is better than doing the wrong thing, but what does it mean to do the right thing?
- We answer this age-old question in an age-old way: by considering the consequences of the agent's behavior.
  - When an agent is plunked down in an environment, it generates a sequence of actions according to the percepts it receives.
  - This sequence of actions causes the environment to go through a sequence of states.
  - If the sequence is desirable, then the agent has performed well.
  - This notion of desirability is captured by a performance measure that **PERFORMANCE MEASURE** evaluates any given sequence of environment states.
- *As a general rule, it is better to design performance measures according to what one actually wants in the environment, rather than according to how one thinks the agent should behave.*

How Agents should act?

- A rational agent should strive to "do the right thing", based on what it can perceive and the actions it can perform.
  - What does right thing mean? one that will cause the agent to be most successful and is expected to maximize goal achievement, given the available information
- A rational agent is not omniscient
  - An Omniscient agent knows the actual outcome of its actions, and can act accordingly, but in reality omniscience (being infinitely wise) is impossible.
  - Rational agents take action with expected success, where as omniscient agent take action with 100% certainty of success
- Are human beings Omniscient or Rational agent?

Example: Is the agent Rational?

- Megersa was walking along the road to Bus Station; He saw an old friend across the street. There was no traffic.
- So, being rational, he started to cross the street.
- Meanwhile a big banner falls off from above and before he finished crossing the road, he was flattened.
- Was Megersa irrational to cross the street?
- This points out that rationality is concerned with expected success, given what has been perceived.
  - Crossing the street was rational, because most of the time, the crossing would be successful, and there was no way you could have foreseen the falling banner.
  - The EXAMPLE shows that we can not blame an agent for failing to take into account something it could not perceive. Or for failing to take an action that it is incapable of taking.

Omniscience, learning, and autonomy

- We need to be careful to distinguish between rationality and omniscience.
- An omniscient agent knows the actual outcome of its actions and can act accordingly; but omniscience is impossible in reality.
- Rationality is not the same as perfection. Rationality maximizes expected performance, while perfection maximizes actual performance.
- Our definition of rationality does not require omniscience, then, because the rational choice depends only on the percept sequence to date.
- Doing actions in order to modify future percepts—sometimes called information gathering—is an important part of rationality.
- A second example of information gathering is provided by the exploration that **EXPLORATION** must be undertaken by a vacuum-cleaning agent in an initially unknown environment.

Omniscience, learning, and autonomy

- Our definition requires a rational agent not only to gather information but also to **learn** as much as possible from what it perceives.
- The agent's initial configuration could reflect some prior knowledge of the environment, but as the agent gains experience this may be modified and augmented.
- There are extreme cases in which the environment is completely known a priori.
  - In such cases, the agent need not perceive or learn; it simply acts correctly.
  - Of course, such agents are fragile.
  - EXAMPLES: Consider the lowly dung beetle. How it move it?

Omniscience, learning, and autonomy

- To the extent that an agent relies on the prior knowledge of its designer rather than on its own percepts, we say that the agent lacks autonomy.
- A rational agent should be autonomous—it should learn what it can to compensate for partial or incorrect prior knowledge.
  - For example, a vacuum-cleaning agent that learns to foresee where and when additional dirt will appear will do better than one that does not.
- As a practical matter, one seldom requires complete autonomy from the start: when the agent has had little or no experience, it would have to act randomly unless the designer gave some assistance.
- After sufficient experience of its environment, the behavior of a rational agent can become effectively independent of its prior knowledge.
  - Hence, the incorporation of **learning** allows one to design a single rational agent that will succeed in a vast variety of environments.

Acting of Intelligent Agents (Rationality)

- In summary what is rational at any given point depends on PEAS (Performance measure, Environment, Actuators, Sensors) framework.
  - Performance measure
    - The performance measure that defines degrees of success of the agent
  - Environment
    - Knowledge: What an agent already knows about the environment
  - Actuators – generating actions
    - The actions that the agent can perform back to the environment
  - Sensors – receiving percepts
    - Perception: Everything that the agent has perceived so far concerning the current scenario in the environment
- 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.

Performance measure

- How do we decide whether an agent is successful or not?
  - Establish a standard of what it means to be successful in an environment and use it to measure the performance
  - If you can't measure performance, then you can not tell whether an agent is successful or not.
  - A rational agent should do whatever action is expected to maximize its performance measure, on the basis of the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

Example: PEAS

- Consider the task of designing an automated taxi driver agent:
  - Performance measure: Safe, fast, legal, comfortable trip, shortest distance, maximize profits
  - Environment: Roads, other traffic, pedestrians, customers
  - Actuators: Artificial legs & hands, Speaker
  - Sensors: Cameras, GPS, engine sensors, recorder (microphone)
  - Goal: driving safely from source to destination point

Examples: Agents for Various

Agent type	Percepts	Actions	Goals	Environment
Interactive English tutor	Typed words, Keyboard	Print exercises, suggestions, corrections	Maximize student's score on test	Set of students
Medical diagnosis system	Symptoms, patient's answers	Questions, tests, treatments	Healthy person, minimize costs	Patient, hospital
Part-picking robot	Pixels of varying intensity	Pick up parts and sort into bins	Place parts in correct bins	Conveyor belts with parts
Satellite image analyser	Pixels of varying intensity, color	Print a categorization of scene	Correct categorization	Images from orbiting satellite
Refinery controller	Temperature, pressure readings	Open, close valves; adjust temperature	Maximize purity, yield, safety	Refinery

Classes of Environments

- Actions are done by the agent on the environment. Environments provide percepts to an agent.
- Agent perceives and acts in an environment. Hence in order to design a successful agent , the designer of the agent has to understand the type of the environment it interacts with.
- Properties of Environments:
  - Fully observable vs. Partially observable
  - Deterministic vs. Stochastic(random)
  - Episodic vs. Sequential
  - Static vs. Dynamic
  - Discrete vs. Continuous
  - Single agent vs. Multi agent

Fully observable vs. partially observable

- Does the agent's sensory see the complete state of the environment?
  - If an agent has access to the complete state of the environment, then the environment is accessible or fully observable.
  - If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is fully observable.
  - Fully observable environments are convenient because the agent need not maintain any internal state to keep track of the world.
- An environment is effectively accessible if the sensors detect all aspects that are relevant to the choice of action.
- Any example of fully observable
  - In a chess game, the state of the system, that is, the position of all the players on the chess board, is available the whole time so the player can make an optimal decision?

Fully observable vs. partially observable

- 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—for example, a vacuum agent with only a local dirt sensor cannot tell whether there is dirt in other squares, and an automated taxi cannot see what other drivers are thinking.
- If the agent has no sensors at all then the environment is UNOBSERVABLE, but, the agent's goals may still be achievable, sometimes with certainty.
- Taxi driving is partially observable

Single-agent vs. Multi-agent

- If an agent operate by itself in an environment, it is a single agent environment.
- For example, an agent solving a crossword puzzle by itself is clearly in a single-agent environment, whereas an agent playing chess is in a two-agent environment.
  - How do you decide whether another entity must be viewed as an agent?
  - Key question: can its behavior be described as maximizing performance depending on the actions of 'our' agent?
  - Classify multi-agent environment as (partially) competitive and/or (partially) cooperative
  - Ex: Taxi is partially competitive and partially cooperative

Single-agent vs. Multi-agent

- For example, in chess which has A & B player, the opponent entity B is trying to maximize its performance measure, which, by the rules of chess, minimizes agent A's performance measure.
- Thus, chess is a competitive multi-agent environment. In the taxi-driving environment, on the other hand, avoiding collisions maximizes the performance measure of all agents, so it is a partially cooperative multi-agent environment. It is also partially competitive because, for example, only one car can occupy a parking space.
- The agent-design problems in multi-agent environments are often quite different from those in single-agent environments;
  - For example, communication often emerges as a rational behavior in multi-agent environments; in some competitive environments, randomized behavior is rational because it avoids the pitfalls of predictability.

Deterministic vs. stochastic

- Is there a unique mapping from one state to another state for a given action?
- The environment is deterministic if the next state is completely determined by
  - the current state of the environment and
  - the actions selected and executed by the agent;
  - otherwise, it is stochastic
- Taxi driving is non-deterministic (i.e. stochastic)
  - 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, but variations can include stochastic elements such as randomly appearing dirt and an unreliable suction mechanism.

Deterministic vs. stochastic

- We say an environment is uncertain if it is not fully observable or not deterministic.
- One final note: our use of the word "stochastic" generally implies that uncertainty about outcomes is quantified in terms of probabilities; a nondeterministic environment is one in which actions are characterized by their possible outcomes, but no probabilities are attached to them.
- Nondeterministic environment descriptions are usually associated with performance measures that require the agent to succeed for all possible outcomes of its actions.

Episodic vs. Sequential

- Does the next “episode” or event depend on the actions taken in previous episodes? No
- In an episodic environment, the agent's experience is divided into "episodes".
  - Each episode consists of the agent perceiving and then performing a single action, and the choice of action in each episode depends only on the episode itself.
  - The quality of its action depends just on the episode itself.
  - For example, an agent that has to spot defective parts on an assembly line.
- In sequential environment the current decision could affect all future decisions
  - Chess and taxi driving are sequential: in both cases, short-term actions can have long-term consequences.

Episodic vs. Sequential

- Does the next “episode” or event depend on the actions taken in previous episodes? No
- In an episodic environment, the agent's experience is divided into "episodes".
  - Each episode consists of the agent perceiving and then performing a single action, and the choice of action in each episode depends only on the episode itself.
  - The quality of its action depends just on the episode itself.
  - For example, an agent that has to spot defective parts on an assembly line.
- In sequential environment the current decision could affect all future decisions
  - Chess and taxi driving are sequential: in both cases, short-term actions can have long-term consequences.

Static vs. Dynamic

- Can the world change while the agent is thinking and on purpose?
  - If the environment can change while the agent is on purpose, then we say the environment is dynamic for that agent
  - otherwise it is static.
- Static environments are easy to deal with because the agent need not keep looking at the world while it is deciding on an action, nor need it worry about the passage of time.
- Dynamic environments, on the other hand, are continuously asking the agent what it wants to do; if it hasn't decided yet, that counts as deciding to do nothing.
- Examples:
  - Taxi driving is dynamic
  - Crossword puzzles are static.

Static vs. Dynamic

- Can the world change while the agent is thinking and on purpose?
  - If the environment can change while the agent is on purpose, then we say the environment is dynamic for that agent
  - otherwise it is static.
- Static environments are easy to deal with because the agent need not keep looking at the world while it is deciding on an action, nor need it worry about the passage of time.
- Dynamic environments, on the other hand, are continuously asking the agent what it wants to do; if it hasn't decided yet, that counts as deciding to do nothing.
- Examples:
  - Taxi driving is dynamic
  - Crossword puzzles are static.

Discrete vs. Continuous

- The discrete/continuous distinction applies to the state of the environment, to the way time is handled, and to the percepts and actions of the agent.
- Are the distinct percepts & actions limited or unlimited?
  - If there are a limited number of distinct, clearly defined percepts and actions, we say the environment is discrete.
  - otherwise it is continuous.
- Examples:
  - Chess also has a discrete set of percepts and actions,
  - Input from digital cameras is discrete,
  - Taxi driving is continuous.

Environment Types

- Below are lists of properties of a number of familiar environments

Problems	Observable	Deterministic	Episodic	Static	Discrete	Single agent
Crossword Puzzle	Yes	Yes	No	Yes	Yes	Yes
Part-picking robot	No	No	Yes	No	No	No
Web shopping program	No	No	No	No	Yes	No
Tutor	No	No	No	Yes	Yes	Yes
Medical Diagnosis	No	No	No	No	No	No
Taxi driving	No	No	No	No	No	No

Hardest case: an environment that is inaccessible, sequential, stochastic, dynamic, continuous and multi-agent, which is true in the real world.

Structure of Intelligent Agents

Designing an agent

- The job of AI is to design an agent program that implements the agent function—the mapping from percepts to actions.
- A physical agent has two parts: architecture + program
- Architecture
  - Runs the programs
  - Makes the percept from the sensors available to the programs
  - Feeds the program's action choices to the effectors
- Programs
  - Accepts percept from an environment and generates actions
    - Before designing an agent program, we need to know the possible percept and actions
  - By enabling a learning mechanism, the agent could have a degree of autonomy, such that it can reason and take decision

Agent Types

- Five basic types in order of increasing generality:
  - Simple reflex agent
  - Model-based reflex agent
  - Goal-based agent
  - Utility-based agent
  - Learning agents

Simple reflex agents

- These agents select actions on the basis of the current percept, ignoring the rest of the percept history.
- It works by finding a rule whose condition matches the current situation (as defined by the percept) and then doing the action associated with that rule.

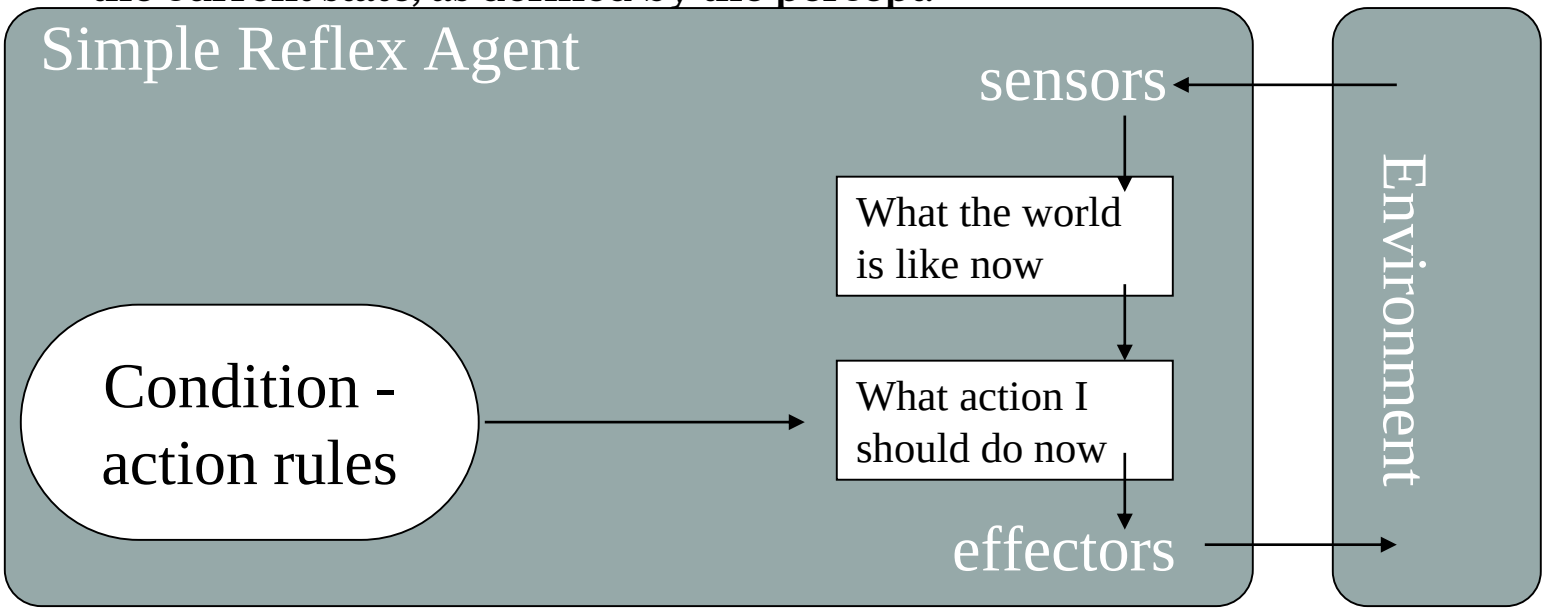
E.g. If the car in front brakes, and its brake lights come on, then the driver should notice this and initiate braking,

  - Some processing is done on the visual input to establish the condition. If "The car in front is braking"; then this triggers some established connection in the agent program to the action "initiate braking". We call such a connection a condition-action rule written as: If car-in-front-is breaking then initiate-braking.

- Humans also have many such conditions. Some of which are **learned responses** while others are **innate (inborn) responses**
  - Ex. Blinking when something approaches the eye.

Structure of a simple reflex agent

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



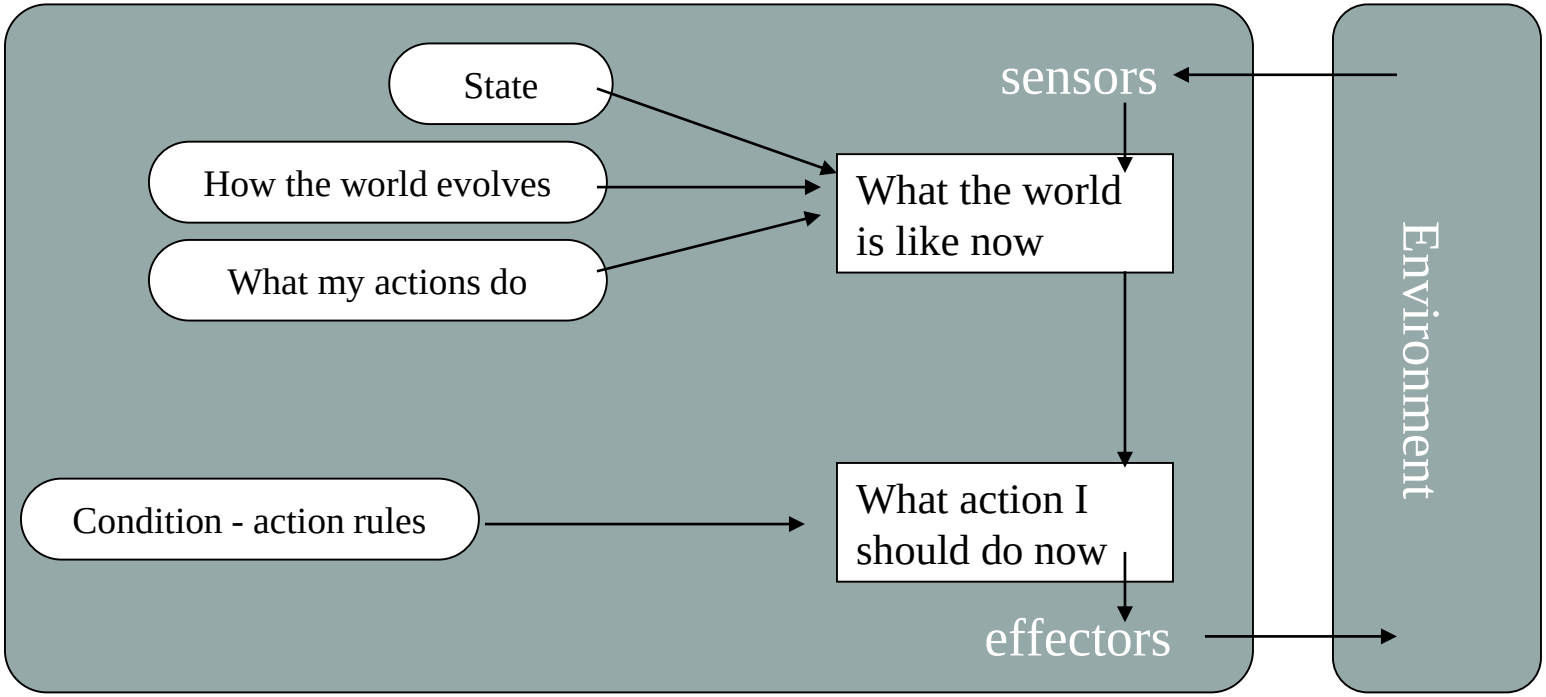
function SIMPLE-REFLEX-AGENT(*percept*) **returns** action
  static: *rules*, a set of condition-action rules

  state ← INTERPRET-INPUT (*percept*)
  rule ← RULE-MATCH (*state*,*rules*)
  action ← RULE-ACTION [*rule*]
  return *action*

Model-Based Reflex Agent

- This is a reflex agent with internal state.
  - It keeps track of the world that it can't see now.
  - It works by finding a rule whose condition matches the current situation (as defined by the percept and the stored internal state)
  - If the car is a recent model -- there is a centrally mounted brake light. With older models, there is no centrally mounted, so what if the agent gets confused?
  - Is it a parking light? Is it a brake light? Is it a turn signal light?
  - Some sort of internal state should be in order to choose an action.
- The camera should detect two red lights at the edge of the vehicle go ON or OFF simultaneously.
- The driver should look in the rear-view mirror to check on the location of near by vehicles. In order to decide on lane-change the driver needs to know whether or not they are there.
  - The driver sees, and there is already stored information, and then does the action associated with that rule.

Structure of Model-Based reflex agent



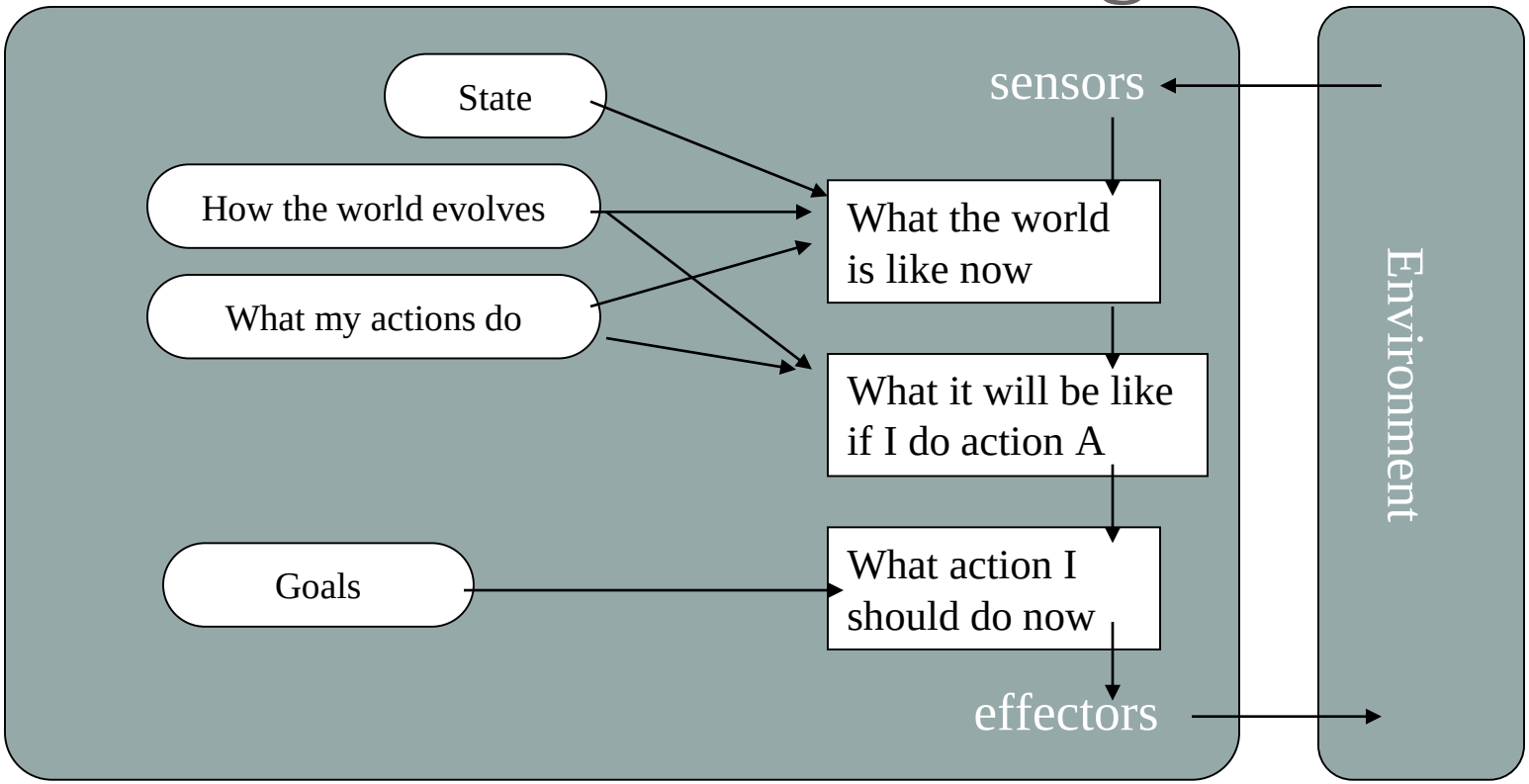
function REFLEX-AGENT-WITH-STATE (*percept*) **returns** action
static: *state*, a description of the current world state
rules, a set of condition-action rules

state ← UPDATE-STATE (*state*, *percept*)
rule ← RULE-MATCH (*state*, *rules*)
action ← RULE-ACTION [*rule*]
state ← UPDATE-STATE (*state*, *action*)
return *action*

Goal based agents

- Choose actions that achieve the goal (an agent with explicit goals)
- Involves consideration of the future:
  - Knowing about the current state of the environment is not always enough to decide what to do.
- For example, at a road junction, the taxi can turn left, right or go straight.
  - The right decision depends on where the taxi is trying to get to. As well as a current state description, the agent needs some sort of goal information, which describes situations that are desirable. E.g. being at the passenger's destination.
- The agent may need to consider long sequences, twists and turns to find a way to achieve a goal.

Structure of a Goal-based agent

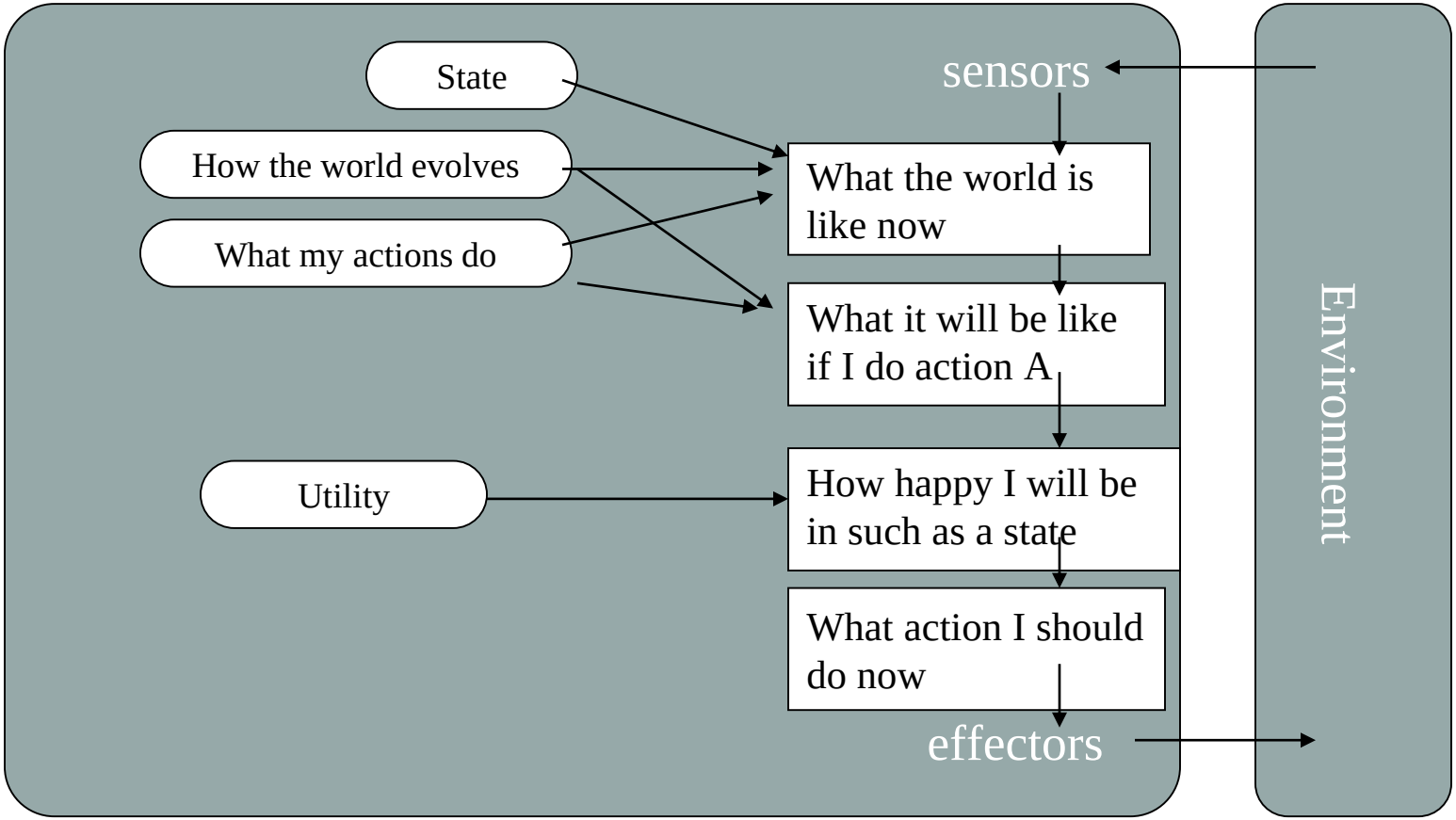


function GOAL_BASED_AGENT (*percept*) **returns** action
  state ← UPDATE-STATE (*state*, *percept*)
  action ← SELECT-ACTION [*state*, *goal*]
  state ← UPDATE-STATE (*state*, *action*)
 return *action*

Utility based agents

- Goals are not really enough to generate high quality behavior.
  - For e.g., there are many action sequences that will get the taxi to its destination, thereby achieving the goal. Some are quicker, safer, more reliable, or cheaper than others. We need to consider Speed and safety
- When there are several goals that the agent can aim for, non of which can be achieved with certainty. Utility provides a way in which the likelihood of success can be weighed up against the importance of the goals.
- An agent that possesses an explicit utility function can make rational decisions.

Structure of a utility-based agent



function UTILITY_BASED_AGENT (*percept*) **returns** action

state ← UPDATE-STATE (*state*, *percept*)
action ← SELECT-OPTIMAL_ACTION [*state*, *goal*]
state ← UPDATE-STATE (*state*, *action*)
return *action*

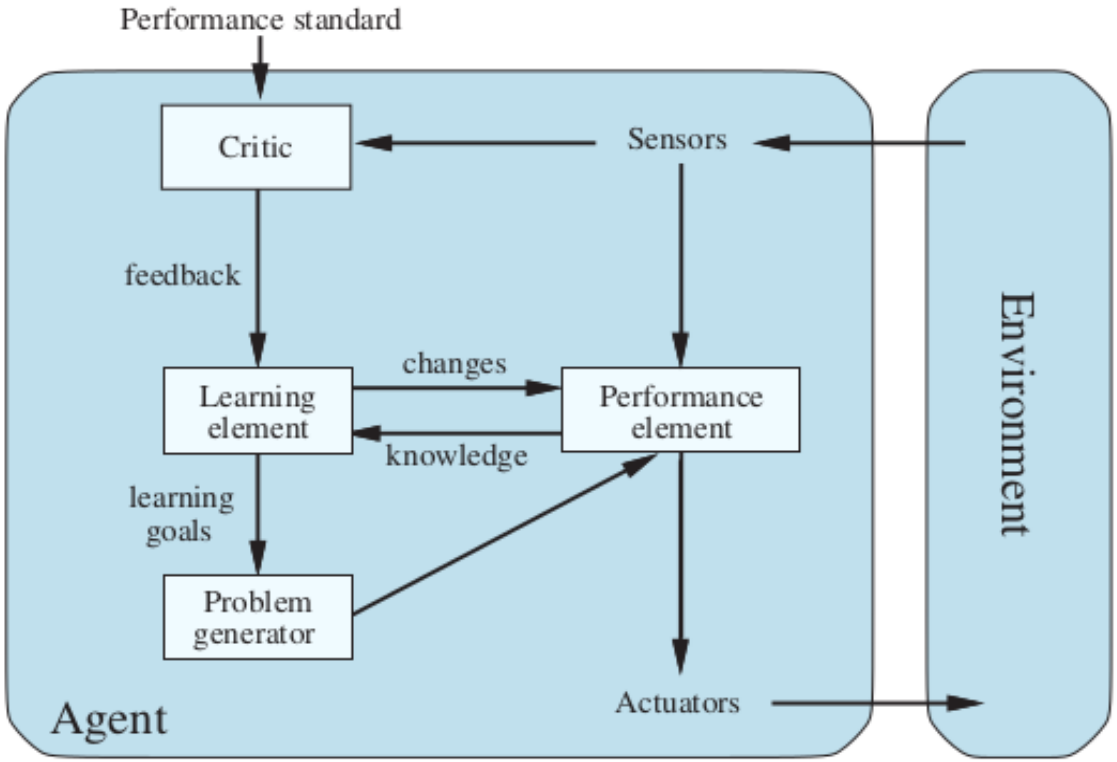
Learning agents

- A learning agent can be divided into four conceptual components.
- The most important distinction is between the learning element, which is responsible for making improvements, and the performance element, which is responsible for selecting external actions.
- The **performance element** is what we have previously considered to be the entire agent: it takes in percepts and decides on actions.
- The **learning element** uses feedback from the critic on how the agent is doing and determines how the performance element should be modified to do better in the future.
- The design of the learning element depends very much on the design of the performance element.
- Given a design for the performance element, learning mechanisms can be constructed to improve every part of the agent.

Learning agents

- The **critic** tells the learning element how well the agent is doing with respect to a fixed performance standard. The critic is necessary because the percepts themselves provide no indication of the agent's success. It is important that the performance standard be fixed.
- The last component of the learning agent is the **problem generator**. It is responsible for suggesting actions that will lead to new and informative experiences.
- Improving the model components of a model-based agent so that they conform better with reality is almost always a good idea, regardless of the external performance standard. In a sense, the performance standard distinguishes part of the incoming percept as a reward (or penalty) that provides direct feedback on the quality of the agent's behavior. More generally, human choices can provide information about human preferences.
- Learning in intelligent agents can be summarized as a process of modification of each component of the agent to bring the components into closer agreement with the available feedback information, thereby improving the overall performance of the agent.

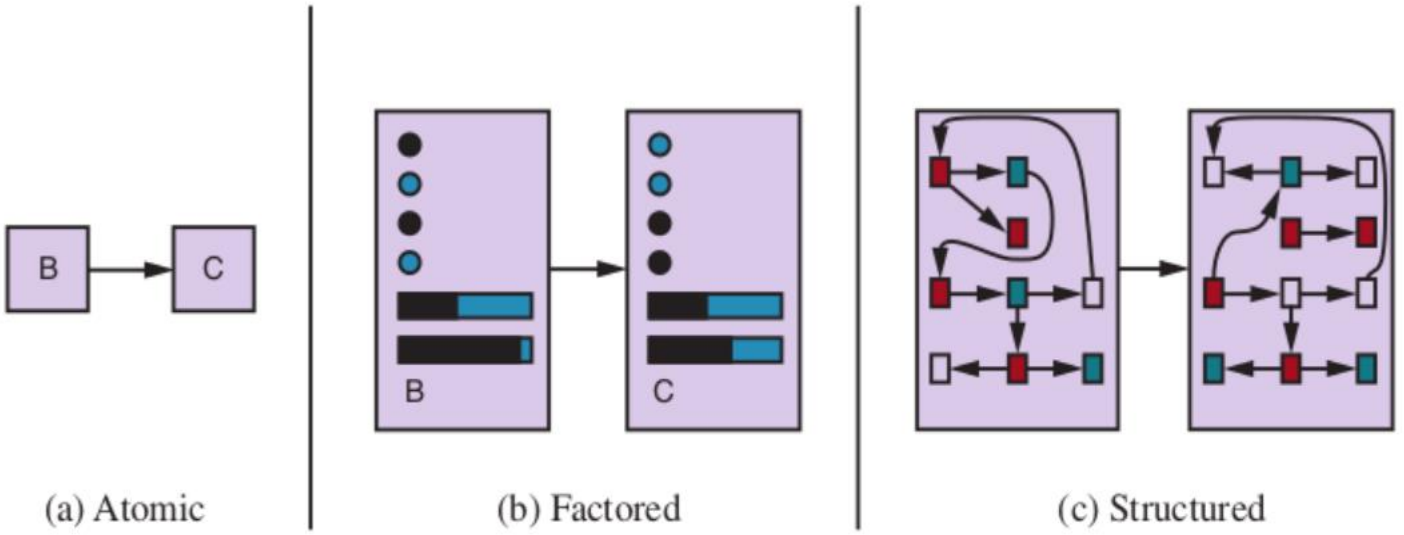
Structure of a Learning agents



A general learning agent. The "performance element" box represents what we have previously considered to be the whole agent program. Now, the "learning element" box gets to modify that program to improve its performance.

How the components of agent programs work

- Roughly speaking, we can place the representations along an axis of increasing complexity and expressive power—atomic, factored, and structured.



Three ways to represent states and the transitions between them. (a) Atomic representation: a state (such as B or C) is a black box with no internal structure; (b) Factored representation: a state consists of a vector of attribute values; values can be Boolean, real-valued, or one of a fixed set of symbols. (c) Structured representation: a state includes objects, each of which may have attributes of its own as well as relationships to other objects.

How the components of agent programs work

- In an atomic representation each state of the world is indivisible—it has no internal structure. The standard algorithms underlying search and game-playing, hidden Markov models, and Markov decision processes all work with atomic representations.
- A factored representation splits up each state into a fixed set of variables or attributes, each of which can have a value. Many important areas of AI are based on factored representations, including constraint satisfaction algorithms, propositional logic, planning, Bayesian networks, and various machine learning algorithms.
- In a Structured representation, objects and their various and varying relationships can be described explicitly. Structured representations underlie relational databases and first-order logic, first-order probability models, and much of natural language understanding.

How the components of agent programs work

- The axis along which atomic, factored, and structured representations lie is the axis of increasing expressiveness. Roughly speaking, a more expressive representation can capture, at least as concisely, everything a less expressive one can capture, plus some more.
- Another axis for representation involves the mapping of concepts to locations in physical memory, whether in a computer or in a brain. If there is a one-to-one mapping between concepts and memory locations, we call that a localist representation. On the other hand, if the representation of a concept is spread over many memory locations, and each memory location is employed as part of the representation of multiple different concepts, we call that a distributed representation. Distributed representations are more robust against noise and information loss.

The End of Topic 2