CS 3011: Artificial Intelligence

Introduction

Instructors: Dr. Durgesh Singh

CSE Discipline, PDPM IIITDM, Jabalpur -482005

Types of Environments

- An agent may have to act on the following types of environments.
  - Fully observable vs. Partially observable
  - Deterministic vs. Stochastic
  - Episodic vs. Sequential
  - Static vs. Dynamic
  - Discrete vs. Continuous
  - Single agent vs. multiagent

Fully Observable vs. Partially Observable

- An environment is fully observable if an agent's sensors give it access to the complete state of the environment at each point in time.
- For fully observable environment, the sensors should detect all the aspects that are relevant to the choice of action (relevance depends upon the performance measure).
- In fully observable environments the agent need not to maintain internal state to keep the track of the world.
- The environment of crossword puzzle, chess are fully observable.

Fully Observable vs. Partially Observable…

- An environment is partially observable if the entire state of the environment is not known to the agent due to noisy and inaccurate sensors or parts of the state are simply missing from the sensor data.
- For instance, a vacuum cleaner with local dirt sensor cannot tell whether dirt is in other squares or not.
- Similarly, an automated taxi cannot see what the other driver is thinking.
- Part-picking robots, interactive English tutor are some other examples of partially observable.
- If the agent has no sensors at all then the environment is unobservable.

Deterministic Vs. Stochastic

- An environment is deterministic, if the next state of the environment is completely determined by the current state and action executed by the agent.
- An agent need not worry about uncertainty in a fully observable, deterministic environment.
- Cross word puzzle, image analysis are examples of deterministic environments.
- Vacuum cleaner world can also be described as deterministic, but variations can include stochastic elements such as randomly appearing dirt or unreliable suction mechanism, etc.

Deterministic Vs. Stochastic…

- An environment is stochastic, if the next state of the environment is not completely determined by the current state and action executed by the agent.
- Taxi driving is clearly stochastic in the sense, because one can never predict the behavior of the traffic exactly, some faults arise in the taxi without warning, etc.
- Refinery Controller and Interactive English Tutor are also examples of stochastic environments.
- If the environment is deterministic except for the action of the other agents, then the environment is strategic such as a chess or a poker playing agent.

Episodic vs. Sequential

- Episodic (vs. sequential): The agent's experience is divided into atomic "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, e.g., a robot whose job is to detect faulty parts on a line in some factory. Image analysis and part picking robot are also examples of episodic environments.
- In a sequential setting, the next episode depends on the previous one(s), e.g., learning which chess move to execute at each sequential step, in order to win the game at the end
  - Also called a sequential decision process.

Episodic vs. Sequential…

- Chess and taxi driving are sequential; in both cases short term actions have long-term consequences.
- Episodic environments are much simpler than sequential environments because the agent does not need to think ahead.

Static vs. Dynamic

- If the environment can change while an agent is deliberating , then the environment is dynamic; otherwise, it is static.
- Dynamic environments are continuously asking the agent what it wants to do.
- Taxi driving is clearly dynamic as the other cars are also moving while the driving algorithm decides what to do next.
- Static environments are easy to deal with because the agent need not to keep looking at the world.
- Crossword puzzle is an example of static environment.

Static vs. Dynamic…

- If the environment does not change with the passage of time but agent's performance score does, then the environment is semi-dynamic.
- Chess when played with a clock, is an example of semi-dynamic environments.

Discrete vs. Continuous

- The environment is discrete if the number of actions and possible states of the environment is finite otherwise it is continuous.
- A chess game has discrete environment as it has finite number of distinct states and a discrete set of percepts and actions.
- For instance, in case of taxi driving, the speed and location of taxi and other vehicles are continuously changing.
- The taxi driving actions are also continuous such as (steering, angles, etc.)

Single Agent vs. Multi Agent

- If only one agent is involved in the task, then the environment is single agent other wise it is multi agent.
- For instance, A person left alone in a maze is an example of the single-agent system whereas chess or taxi driving is a multi agent scenario.
- A multi agent environment can be competitive or cooperative.

Single Agent vs. Multi Agent...

- An agent is said to be in a competitive environment when it competes against another agent to optimize the output.
  - The game of chess is competitive as the agents compete with each other to win the game which is the output.
- An agent is said to be in a cooperative environment when multiple agents cooperate to produce the desired output.
  - When multiple self-driving cars are found on the roads, they cooperate with each other to avoid collisions and reach their destination which is the output desired.

Environment Types

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

- The environment type largely determines the agent design
- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent.

Agent types

- Four basic types in order of increasing generality:
  - Simple Reflex/ Reactive agents
  - Model-based Reflex / Deliberative agents
  - Goal-based agents
  - Utility-based agents

- And Finally: Learning agents

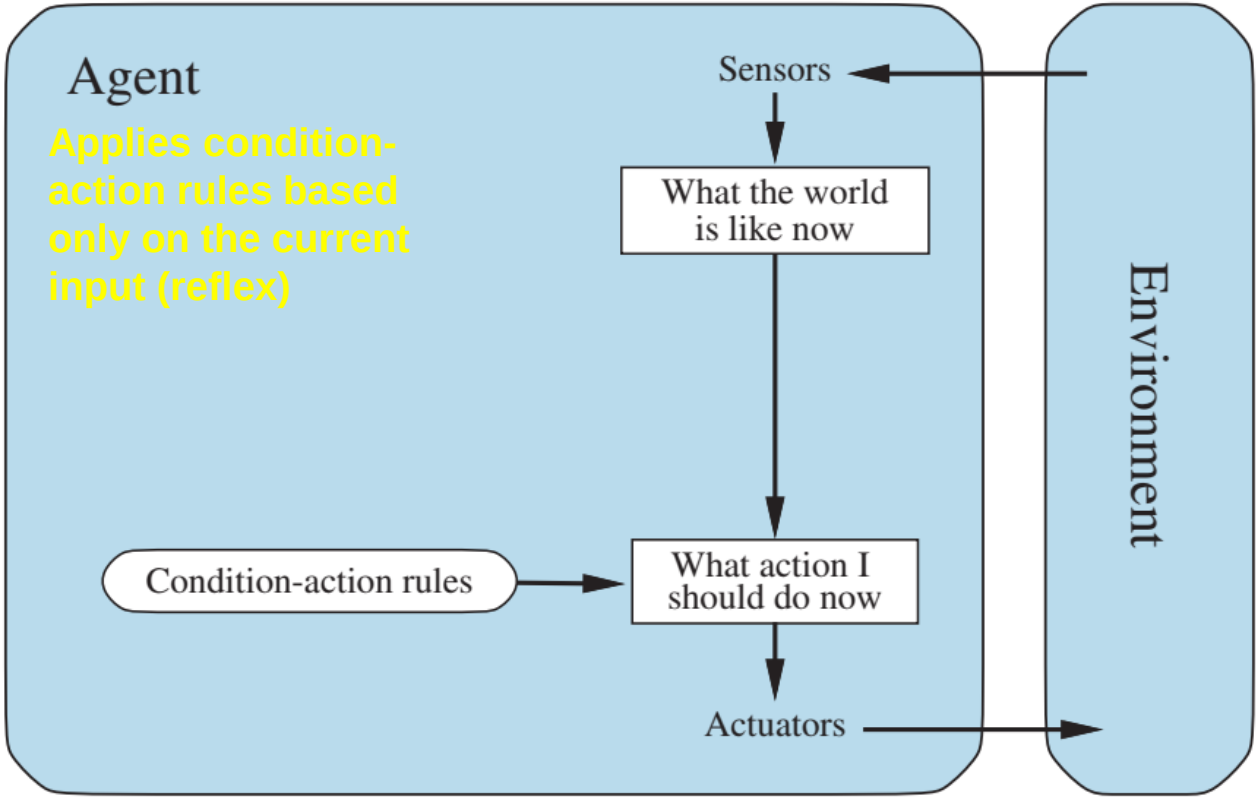
Simple Reflex Agents

- It is the simplest kind of agent.
- The agent selects the action based on the current percept only and ignoring the rest of the percept history.
- It uses just condition-action rules
  - The rules are like the form “if … then …”
- For instance, the simple-reflex program for a vacuum cleaner world is:

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*

Simple Reflex Agents …



Simple-Reflex Agents…

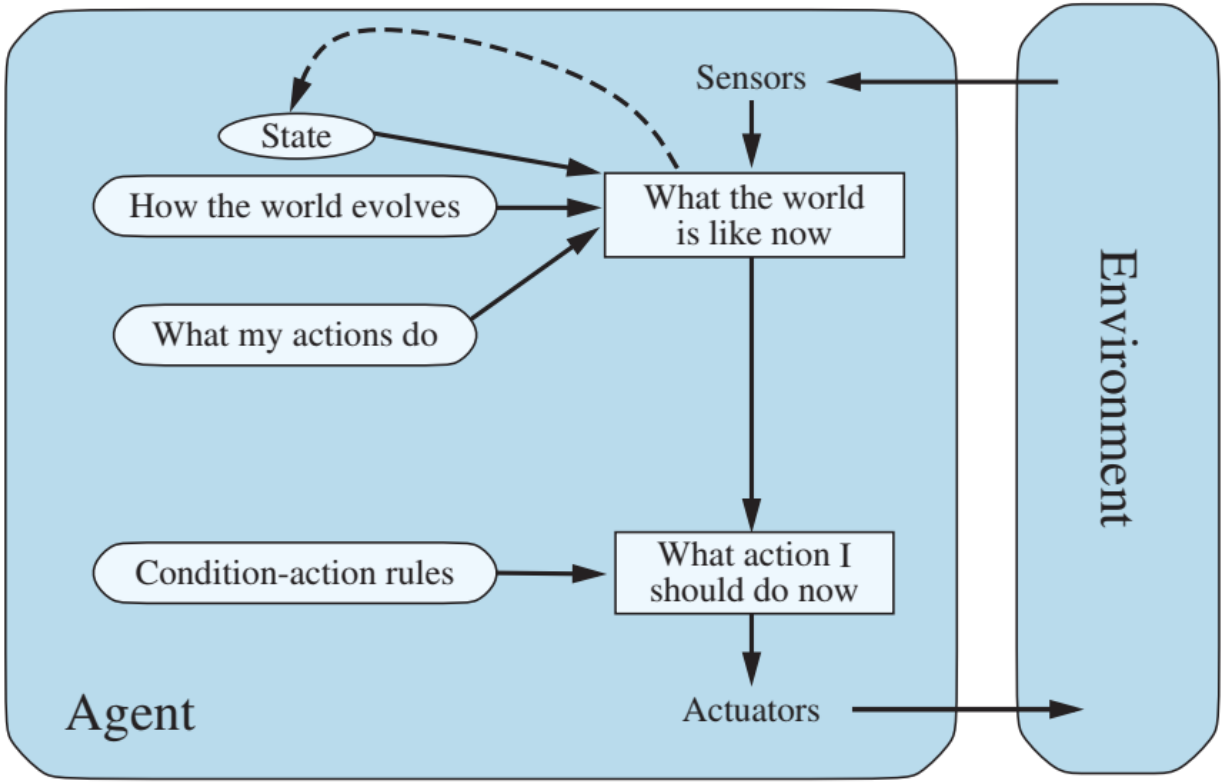
- It works only if the environment is **fully observable**.
- Simple reflex agents operating in partially observable environments, infinite loops are often unavoidable.
- Simple-reflex agents are simple, but they turn out to be of very limited intelligence.

Model-based Reflex agents

- Handle partial observability by keeping track of the part of the world it can't see now.
  - It does this by keeping an internal state that depends on what it has seen before so it holds information on the unobserved aspects of the current state. Ex. driving a car and changing lane.
- Agent keeps track of internal state that depends on the percept history.
- Updating the internal state information as time goes by requires two kinds of knowledge to be encoded in the agent program
  - Information about how the world evolves independently of the agent
  - Information about how the agent's own actions affects the world

Model-based Reflex agents

Handles Partial Observability by creating a model

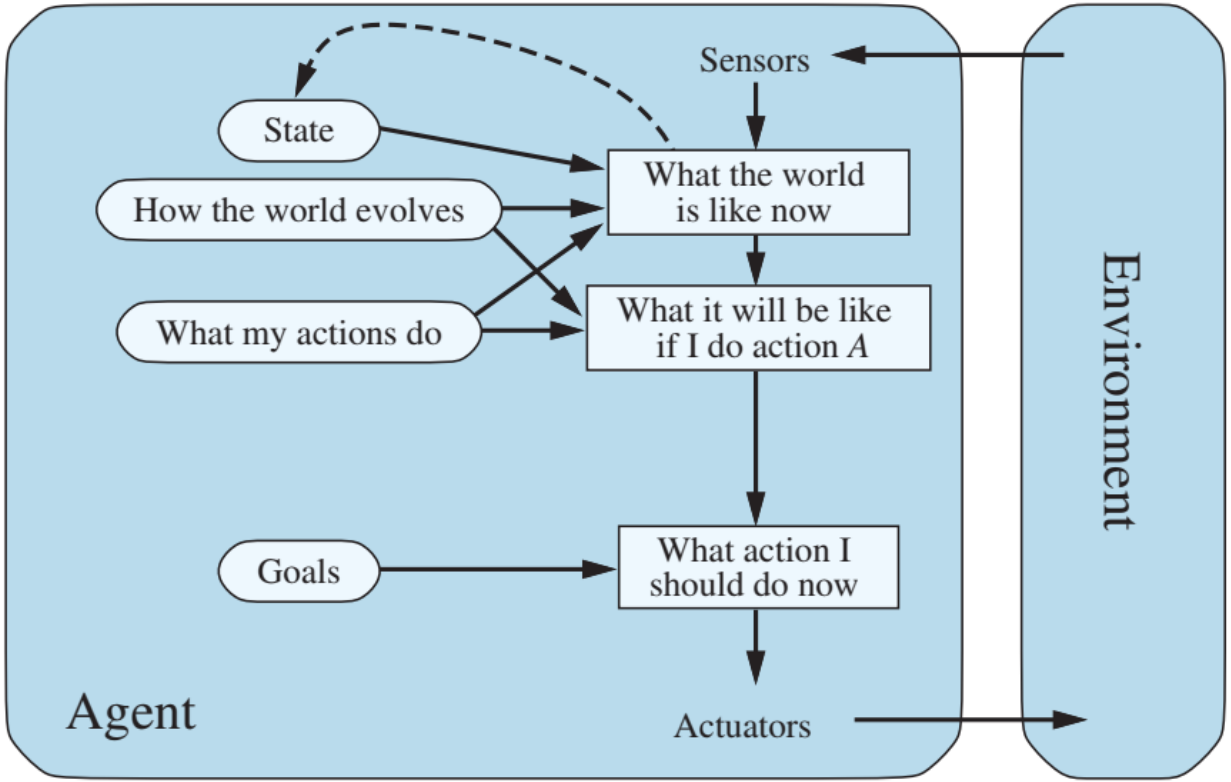


Goal-based agents

- The knowledge of the current state of the environment is not always sufficient.
- In addition to the current state, the agent needs some sort of goal information that describes situations that are desirable.
  - For example, at a road junction, the taxi can turn left, right or go straight on. The correct decision depends upon where the taxi is trying to go i.e., the passenger's destination.
- The agent program must combine the information about the possible actions with the goal information in order to achieve the goal.
- Consider the future with "What will happen if I do A?"

Goal-based agents

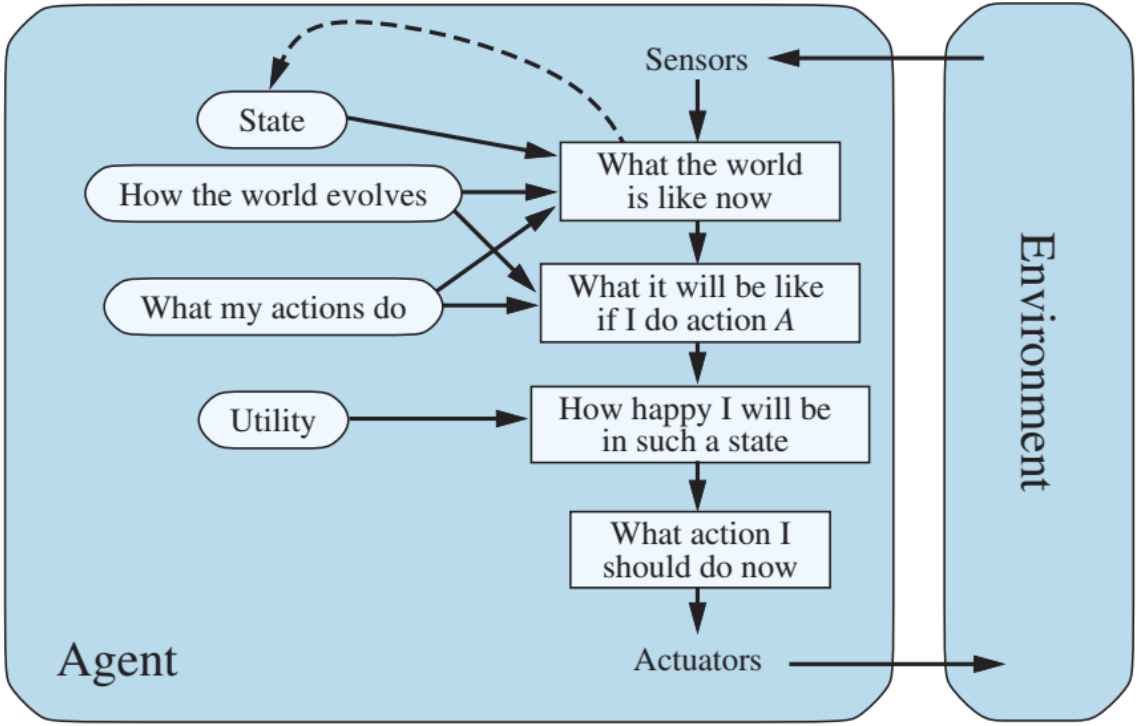
Along with the model, you need goals to direct the agent function.



Utility-based agents

- Sometimes achieving the desired goal is not enough. We may look for quicker, safer, cheaper trip to reach a destination.
- Agent happiness should be taken into consideration. We call it utility.
- A utility function maps a state onto a real number which describes the associated degree of happiness.
- A utility function is the agent's performance measure
- Because of the uncertainty in the world, a utility agent choses the action that maximizes the expected utility

Utility-based agents



Learning Agents

- A learning agent in AI is the type of agent which can learn from its past experiences, or it has learning capabilities.
- It starts to act with basic knowledge and then able to act and adapt automatically through learning.
- A learning agent has mainly four conceptual components, which are:
  - Learning element
  - Critic
  - Performance element
  - Problem generator

Learning Agents

- **Learning element:** It is 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:** It is responsible for selecting external actions. It is what we considered as agent so far.
- **Problem generator:** This component is responsible for suggesting actions that will lead to new and informative experiences.

Learning Agents

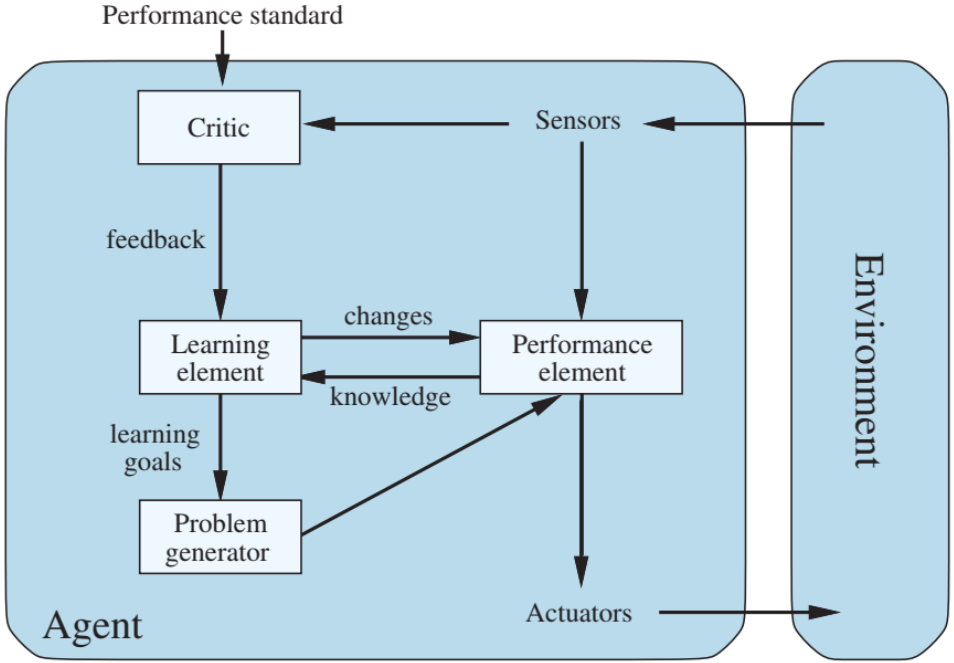


Fig: A general learning agent. The “performance element” box represents what we have previously considered to be the whole agent program. Now, the “learning element” box gets to modify that program to improve its performance.