TDT4136 Introduction to Artificial Intelligence

Chapter 2 - Intelligent Agents

Pinar Öztürk

Norwegian University of Science and Technology

From last week

- Systems that think like humans
- Systems that think rationally
- Systems that act like humans
- Systems that act rationally

Outline

What is an agent?

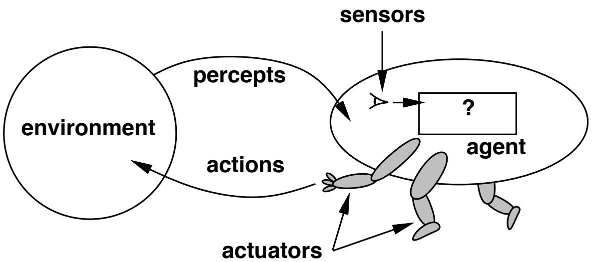
Rationality

PEAS (Performance measure, Environment, Actuators, Sensors)

Environment types

Agent types

Agents and environments



Agent types

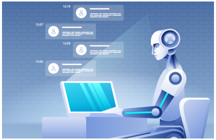
An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators

Agent

Human agent   hardware agent   software agent

(a human driver)

(a robot driver)



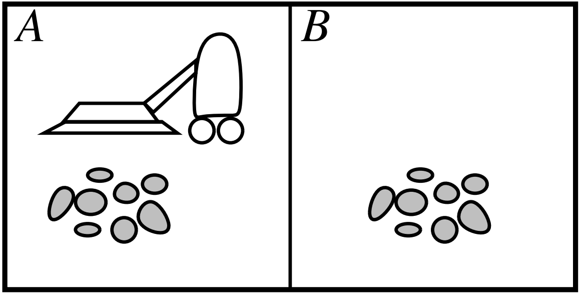
(a chatbot. www.callcentrehelper.com)

Agent

- Human agent: eyes, ears, and other organs for sensors; hands, legs, mouth, and other body parts for actuators

- Robotic agent: cameras and infrared range finders for sensors; various motors for actuators
- Software agent: screen, keyboard, smart phones etc.

Example: Vacuum-cleaner world

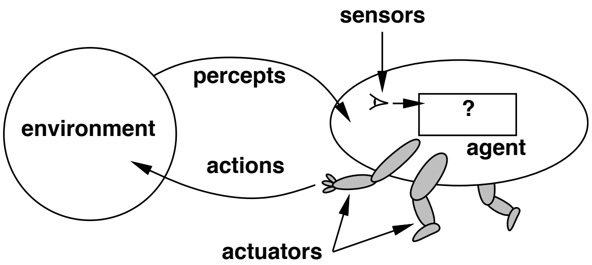


Agent function -and program

An **agent function** maps from percepts to actions:

$f : \mathcal{P} \rightarrow \mathcal{A}$

The **agent program** runs on the physical *architecture* to produce f



Systems that act rationally

- Rational behavior: "doing the right thing". What is the **right thing**?
- May include thinking for action — action without thinking: ?
- Rationality can be defined and optimized
- On the other hand

  - perfect rationality is only feasible in ideal environments.
  - rationality is often not a very good model of reality.
    -humans are, e.g., very bad in estimating probabilities

Rational agent

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

Rational $\implies$ exploration, learning, autonomy

An agent is **autonomous**: takes its decisions (e.g., decide actions) according to the current situation, and changes their "plans" accordingly.

Sphex wasp relies on its innate "plan", hence lacks autonomy.

`https://www.youtube.com/watch?v=YNvi_j2z96w`

Rationality depends on PEAS

P: The *performance* measure

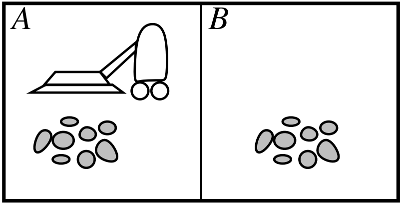
E: The *environment*

A: The *actions* that the agent can perform

S: The percept *sequence*

Design objectives and rationality

- A rational agent performs actions in line with the design objectives.
- These objectives shape the performance measure.
- Selected performance measure evaluates the **environment sequence**
- For example, in case of the vacuum cleaner, the designer could focus only on collecting as much dirt as possible in time T



Design objectives and rationality

- For example, in case of the vacuum cleaner, the designer could focus only on collecting as much dirt as possible in time T
- Or, take also other factors into account, e.g :
  – amount of time taken to clean
  – amount of electricity consumed
  – amount of noise generated, etc
- e.g., one point per square cleaned up in time T, or
  - one point per clean square per time step, minus one per move
  - etc

PEAS

To design a rational agent, we must specify the PEAS

Consider, e.g., designing an automated taxi:

Performance measure??

Environment??

Actuators??

Sensors??

Automated taxi

Performance measure: safety, destination, profits, legality, comfort

Environment: streets/highways, traffic, pedestrians, weather

Actuators: steering, accelerator, brake, horn, speaker/display

Sensors: video, accelerometers, gauges, engine sensors, GPS

Internet shopping agent

Performance measure??

Environment??

Actuators??

Sensors??

Internet shopping agent

Performance measure price, quality, appropriateness, efficiency

Environment current and future WWW sites, vendors, shippers

Actuators display to user, follow URL, fill in form

Sensors HTML pages (text, graphics, scripts)

Medical diagnosis system agent

Performance measure??

Environment??

Actuators??

Sensors??

Medical diagnosis system agent

Performance measure : healthy patient, minimize costs, lawsuits

Environment: patient, hospital, staff

Actuators: screen display (questions, tests, diagnoses, treatments, referrals)

Sensors: keyboard (entry of symptoms, findings, patient's answers)

Properties of environments

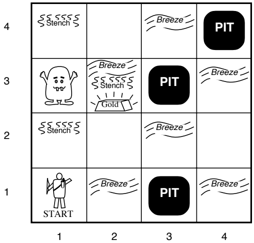
Fully observable environments

- relevant parts of the state of the environment can be sensed
- no need to maintain any internal state to keep track of the world

Partially observable environments

- parts of the environment cannot be sensed
- agent must make informed guesses about world

Fully vs Partially Observable Environments - example



- squares adjacent to Wumpus are smellly
- squares adjacent to Wumpus are beezy
- glitter iff gold is in the same square
- shooting kills wumpus if you are facing it
- only one arrow to shoot

Environment: observable?

Fully Observable vs Partially Observable Environments

Environment	Observable	Deterministic	Episodic	Discrete	Dynamic	Agent
Crossword	fully					
Chess w/clock	fully					
Backgammon	fully					
Taxi driving	partially					
Medical diagnosis	partially					

Properties of environments

Deterministic environment

- Any action has a single guaranteed effect, and no uncertainty/failure.

Stochastic environment

- There is some uncertainty about the outcome of an action
- Multiple outcome alternatives, quantified in terms of probabilities

Environment	Observable	Deterministic	Episodic	Discrete	Dynamic	Agent
Crossword	fully	deterministic				
Chess w/clock	fully	deterministic				
Backgammon	fully	stochastic				
Taxi driving	partially	stochastic				
Medical diagnosis	partially	stochastic				

Properties of environments

Episodic environments

- The agent's experience is divided into atomic episodes.
- Each episode consists of the agent perceiving and then performing a single action
- The choice of action in each episode depends only on the episode itself

Sequential environments

- The current decision could affect all future decisions

Environment	Observable	Deterministic	Episodic	Discrete	Dynamic	Agent
Crossword	fully	deterministic	sequential			
Chess w/clock	fully	deterministic	sequential			
Backgammon	fully	stochastic	sequential			
Taxi driving	partially	stochastic	sequential			
Medical diagnosis	partially	stochastic	sequential			

Properties of environments

Discrete

Finite number of distinct states, percepts and actions,

Continuous

Continuous time/state/actions

Environment	Observable	Deterministic	Episodic	Discrete	Dynamic	Agent
Crossword	fully	deterministic	sequential	discrete		
Chess w/clock	fully	deterministic	sequential	discrete		
Backgammon	fully	stochastic	sequential	discrete		
Taxi driving	partially	stochastic	sequential	continuous		
Medical diagnosis	partially	stochastic	sequential	continuous		

Properties of environments

Dynamic environment

May change while an agent is deliberating

Static environment

The environment does not change

Semidynamic

The world does not change but the agent's performance score may

Environment	Observable	Deterministic	Episodic	Discrete	Dynamic	Agent
Crossword	fully	deterministic	sequential	discrete	static	
Chess w/clock	fully	deterministic	sequential	discrete	semidynamic	
Backgammon	fully	stochastic	sequential	discrete	static	
Taxi driving	partially	stochastic	sequential	continuous	dynamic	
Medical diagnosis	partially	stochastic	sequential	continuous	dynamic	

Properties of environments

Single agent

No other agents - there may be but as a part of the environment

Multi-agent

- Which entities will be viewed as "other agents"?
- The environment contains other agents whose performance measure depends on my actions and vice versa
- Competitive and cooperative interactions

Environment	Observable	Deterministic	Episodic	Discrete	Dynamic	Agent
Crossword	fully	deterministic	sequential	discrete	static	single
Chess w/clock	fully	deterministic	sequential	discrete	semidynamic	multi
Backgammon	fully	stochastic	sequential	discrete	static	multi
Taxi driving	partially	stochastic	sequential	continuous	dynamic	multi
Medical diagnosis	partially	stochastic	sequential	continuous	dynamic	single

Properties of environments

Known environment

- The agent's knowledge about how the environment works/evolves
- Note that a known environment (i.e., the agent knows all the rules that apply) may be only partially observable

Unknown environment

- The agent will have to learn how it works
- An unknown environment can be fully observable.

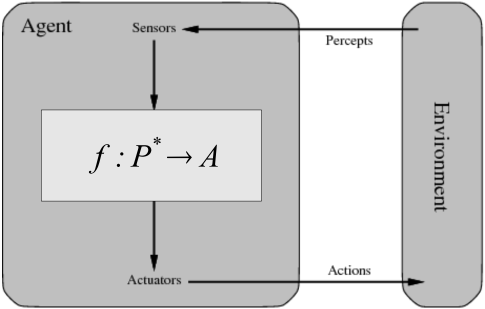
Environment type and agent design

The environment type largely determines the agent design

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

Agent Function

- The agent function will internally be represented by the agent program.
- The agent program runs on the physical architecture to produce f.



A simple vacuum-cleaner agent

Table that maps percept sequences to actions

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

A simple general agent

```
function Table-driven-Agent( [percepts ]) returns an action
    static: table, maps percepts to actions

 append percept to the end of percepts
 action← Lookup(percepts, table)
```

- looks up the right response in the table
- Infeasible: if there are |P| percepts and a life-time of T, then need for a look-up table of size $\sum_{t=1}^{T} |\mathrm{P}|^t$

Agent programs

Writing down such agent functions is not feasible

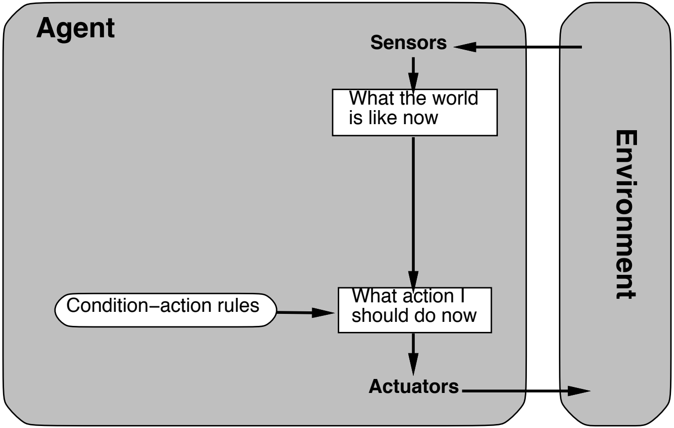
The goal of AI is to write agent programs with small code which produced the desired rational behaviour

Agent types

Four basic types in order of increasing generality:

- – simple reflex agents
- – model-based reflex agent
- – goal-based agents
- – utility-based agents

Simple reflex agents



Simple reflex agent

- Uses only the current percept - ignores the percept sequence,
- Implemented through condition-action rules
- Large reduction in possible percept/action situations from $\sum_{t=1}^{T} |\mathrm{P}|^t$ to $|\mathrm{P}|$

Example:

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*

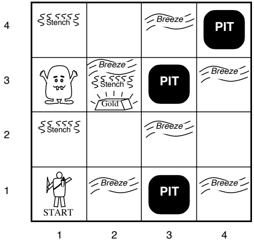
Generic Simple Reflex Agent program

A more general simple reflex agent program:

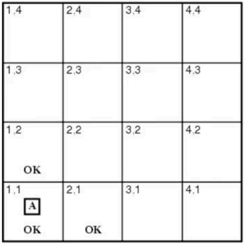
```
function SimpleReflexAgent( percept) returns an action

 persistent rules
 state ← Interpret(percept)
 rule ← Rule-match(state, rules)
 action ← Rule-action(rule)
 return action
```

- Will only work if the environment is fully observable
- everything relevant needs to be determinable from the current input
- otherwise infinite loops may occur, e.g. in the vacuum world without a sensor for the room, the agent does not know whether to move right or left
  - any possible solution: ?

Simple Reflex agent in "Wumpus world" ??



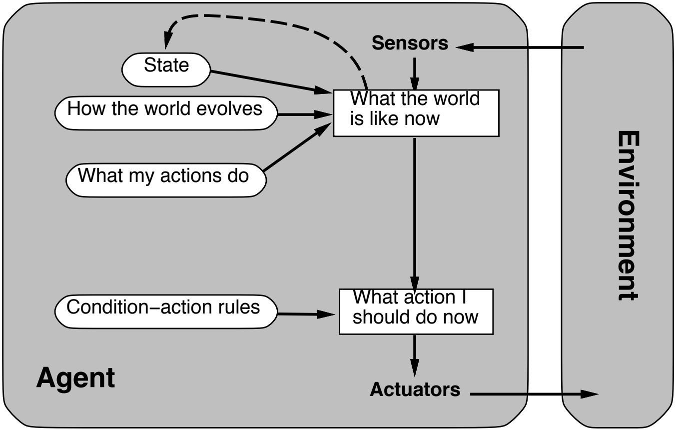
- squares adjacent to Wumpus are smellly
- squares adjacent to Wumpus are beezy
- glitter iff gold is in the same square
- shooting kills wumpus if you are facing it
- only one arrow to shoot



- Observable
  - partial, only local perception
- Deterministic
  - Yes, outcomes are specified
- Episodic
  - No, sequential at the level of actions
- Static
  - Yes, Wumpus and pits do not move
- Discrete
  - Yes
- Single Agent
  - Yes

How good a Simple Reflex (i.e., without memory of past) can do in the Wumpus world?

Model-based Reflex agents



Model-based Reflex agents

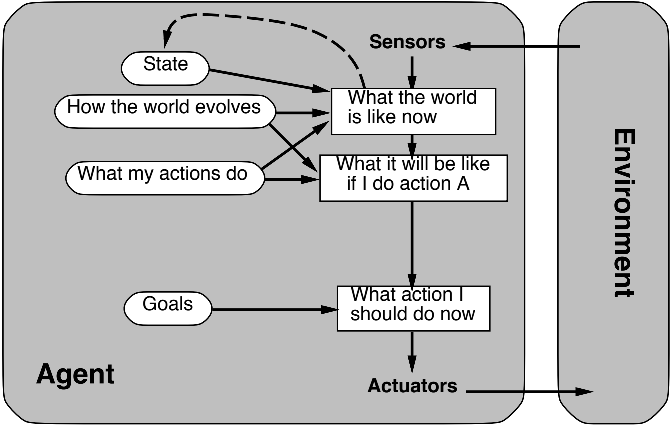
function ModelBasedReflexAgent(*percept*) **returns** an action

persistent *state*, the agent's current conception of the world,
transition-model, a description of how the next state depends on the current state and the action,
sensor-model, a description of how the current world state is reflected in the agent's percepts
rules, a set of condition-action rules
action, the most recent action, initially none

state ← **UpdateState**(*state,action,percept,transition-model,sensor-model*)
rule ← **Rule-match**(*state*, *rules*)
action ← **Rule-action**(*rule*)
return *action*

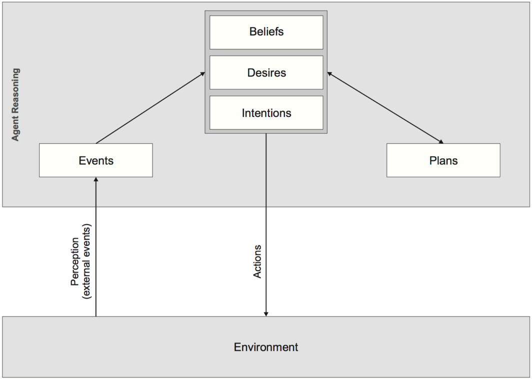
- difficult to exactly determine the current state in partially observable environments - independent from the kind of models used
- hence, may require reasoning under certainty - to "guess" the current situation
- action decision in the same way as the simple reflex agent.

Goal-based agents



Example

- Belief Desire Intention agents (BDI) agents have a **mental state** as the basis for their reasoning.
- Three main mental attitudes: beliefs, desires and, intentions.
- Their reasoning is also called **practical reasoning** - e.g., contrary to deductive reasoning.



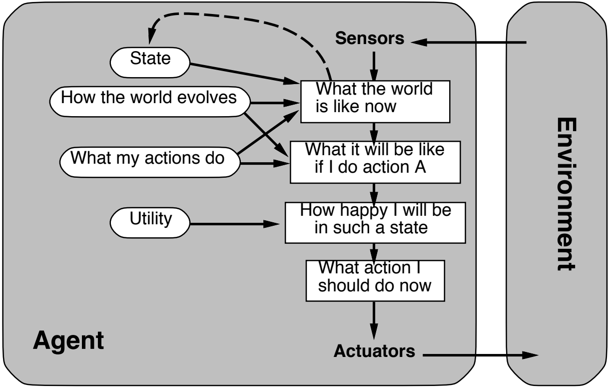
Goal-based agents

- explicit representation of goals: agent knows which states are desirable
- reasoning about goals
- more flexible since knowledge is represented explicitly and can be manipulated
- different reasoning methods than reflex - not condition-action rules and selection of rule/action
- main different from reflex agents: deliberates about future when making decision
- long sequence of actions may be needed to achieve the goal
    - e.g., agents in chapters 3-5 (search) and chapters 10-11 planning
- less efficient than reflex agents - but more flexible

Utility-based agents

Goals provide just a binary happy/unhappy disctinction while utility functions provide a continuous scale

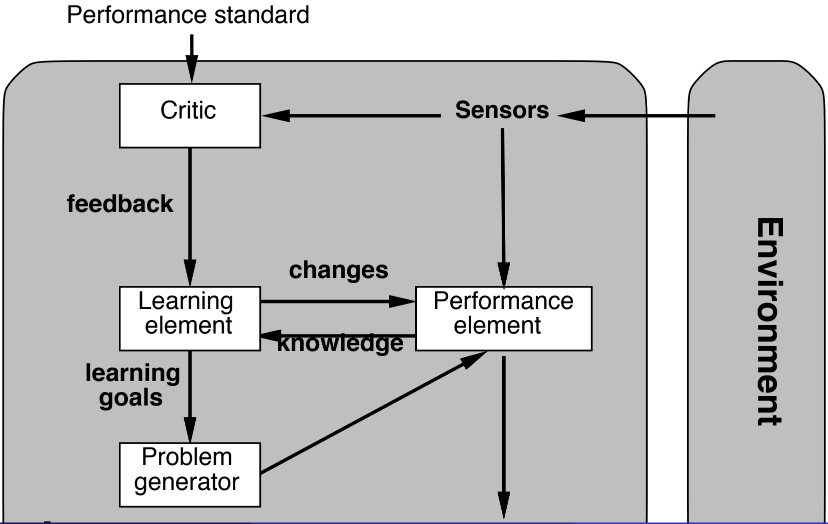
Some goals may be achieved in more than one way, with different utility values



Utility-based agents

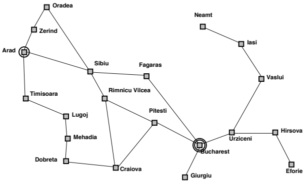
- utilities are internalization of the performance measure
- Utility function maps a state (or a sequence of states) onto a real number
- utilities reflect agent's preferences
- not always know the utility of an action/outcome , hence "expected utility"
- agent chooses actions that maximize the expected utility of the outcomes

Learning agents



Next Week: Goal-based Agents

Goal-based agents that use search methods as Agent Function.
You are at the end of your holiday in Romania. Your return ticket is from Bucharest and you are leaving Arad for Bucharest.



- **Uninformed** Search method as Agent Function.
- In an environment:
  - Fully observable (the agent can see the current state of the world),
  - Deterministic (action has a single outcome)
  - Discrete
  - Known environment (knows which states can be reached through which actions - has a map of Romania.

The week after the next week: **Informed** search methods.