Artificial Intelligence
2 – Intelligent Agents

Arthur Bit-Monnot

INSA 4IR

Section 1

The Agent Model

Intelligent Agents

- we identified the concept of **rational agents**

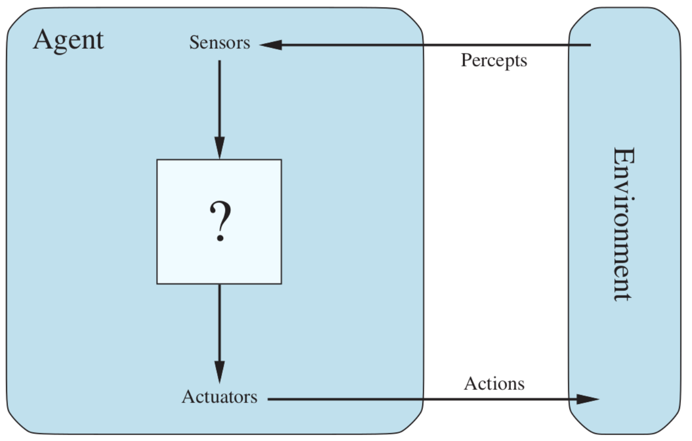
Producing rational agents is the main goal of AI.

- but “intelligence” is not necessary to be rational in a given context

Agent

An **agent** is anything that can be viewed as:

- **perceiving** its environment through **sensors**, and
- **acting** upon that environment through **actuators**



Example Agent classes

Agent	Sensors	Actuators
Human	Eyes, ears, and other organs	Hands, legs, mouth, and other body parts
Robots	Cameras, infrared range finders, force sensors	Various motors, speakers, LEDs
Software agents	Keyboard, mouse, network, file content	Display, file write, network packet send

Environment: the entire the universe!

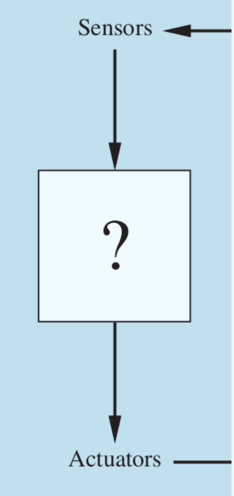
- in practice, we consider a subset of the universe: the one that we can interact with (perceive and affect)

The agent function

The agent's choice of an action at any given instant can depend:

- *on its built-in knowledge*
- *on the entire percept sequence observed to date,*
- *but not on anything it hasn't perceived.*

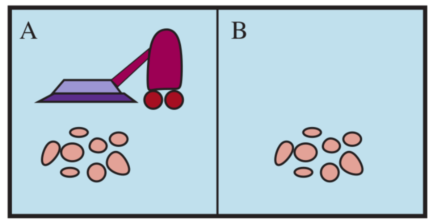
$$agent : Percept^* \rightarrow Action$$



The Vacuum-cleaner World

A vacuum cleaner world with just two locations, A and B.

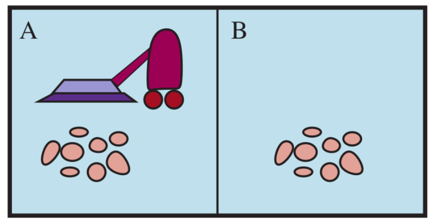
- Each location can be either **Dirty** or **Clean**.
- The agent can:
  - move left or right and
  - suck up dirt in the square it occupies.
- The agent perceives:
  - its location,
  - whether there is dirt in the location.



Agent	Percepts	Actions
Cleaner	A, B, Clean, Dirty	Suck, Left, Right

The Vacuum-cleaner: Tabulated Agent Function

Fully characterized by the *agent function*, mapping each possible percept sequence to an action.



Tabulation of the agent function

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)	Right
...	...

Agent Program

The tabulated agent function is an *external characterization* of the agent's behavior.

- complete characterization of the agent's behavior
- impractical (redundancies, infinite size, . . .)

Agent Program

The tabulated agent function is an *external characterization* of the agent's behavior.

- complete characterization of the agent's behavior
- impractical (redundancies, infinite size, . . .)

Internally, the agent function is represented by an *agent program*.

> *If the current location is dirty, then suck; otherwise, move to the other location.*

Is everything an agent? an AI problem?

- agent is a useful abstraction for thinking about AI problems.
- but almost anything can be seen as an agent.
  - Software is always a mapping from inputs to outputs.

- AI is focused on problems where **non-trivial decisions** are to be made (i.e. that require some form of intelligence)
- AI is interested in methods that have some **generality**.

Those notions are not well defined, and many people will disagree on what perrtains to AI.

Section 2

Rationality

What's a good behavior?

Consequentialism: an agent's behavior is evaluated based on the consequences of its actions.

- actions cause environment to go through a sequence of states.
- if the sequence is "desirable", the agent performed well (rational)
- desirability is defined by a **performance measure**.

Humans: desire and preferences of our own

- rationality: success in choosing actions that move the environment through a trajectory **desirable from their point of view**.

Artificial Agents: no own desires/preferences

- need to be build purpose into the machine (explicitly or implicitly)

Which performance metric for the vacuum cleaner?

- What are the consequences of its actions?
- What objective would you give it?
- What performance measure would you use?

Replace Me Game

Performance Measure for a computer science teacher agent

- What are the consequences of its actions?
- What objective would you give it?
- What performance measure would you use?

Replace Me Game: correction elements

Consequences:

- knowledge and skills of students (available long term)
  - indirect impact: which job you get (fulfilling, well payed), how you reflect on your activity
  - indirect impact: enable you to
    - solve new problems
    - solve problems more efficiently
    - learn new things
- mental health of students (for the course duration)

Objective:

- make the world a better place, within the constraints of the job (I am a public servant)
  - contribute to the happiness of students
  - increase the contribution of students to society
    - economic, social, cultural, ...
- operate in a lewful, socially acceptable way

Replace Me Game: correction elements

Performance measures:

- grades of students? (no, I am the one making the exam)
- grades of students on an external exam? (risk of over specialization, TOEIC)
- evaluation of the course by students? (shortsighted, easy to optimize for)
- number of students burning out, filling formal complaint? (yes, should always be avoided but more a constraint than an objective)
- salary of the students after graduation? (interesting, let the market evaluate but very indirect and shared causality and focus on economic value)
- number of students that work in the field after graduation? (interesting, correlates both with interest of students in the course/fields and society's demand)
- **how to combine them?** (accross metrics, accross students)

Rational Agent Definition

*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.*

Omniscience

Can a rational agent make decisions that are bad in hindsight?

(crossing the street example)

Omniscience

Can a rational agent make decisions that are bad in hindsight?

(crossing the street example)

- yes, the actual outcome of an action is not always predictable
- rationality does not require omniscience
- rationality is about making the best decision given the information available at the time of the decision

Information gathering: doing actions to modify future percepts

Before crossing the street, a rational agent would look both ways to gather information about the traffic.

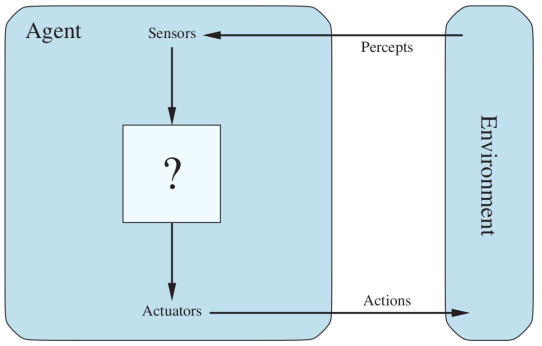
Section 3

Nature of Environments

Nature of Environments

Task environment: the specific problem to be solved

- **P**erformance measure
- **E**nvironment
- **A**ctuators
- **S**ensors



***What are the properties of the environment** that are relevant to the design of an intelligent agent?*

Fully vs Partially Observable

- **Fully observable**: the agent's sensors give it access to the complete state of the environment at each point in time.
  - the agent knows the state of the environment (without bookeeping)
  - no uncertainty about the environment
  - limited to the relevant part of the environment

Fully vs Partially Observable

- **Fully observable**: the agent's sensors give it access to the complete state of the environment at each point in time.
  - the agent knows the state of the environment (without bookeeping)
  - no uncertainty about the environment
  - limited to the relevant part of the environment
- **Partially observable**: the agent's sensors give it only partial information about the state of the environment.
  - blind to some aspects
  - noise in the sensors

Fully vs Partially Observable

- **Fully observable**: the agent's sensors give it access to the complete state of the environment at each point in time.
  - the agent knows the state of the environment (without bookeeping)
  - no uncertainty about the environment
  - limited to the relevant part of the environment
- **Partially observable**: the agent's sensors give it only partial information about the state of the environment.
  - blind to some aspects
  - noise in the sensors
- **Unobservable**: no sensor input, relies on prior knowledge only

Single vs Multi-agent

- **Single agent**: the agent is the only entity acting in the environment
- **Multi-agent**: several agents . . .

Single vs Multi-agent

- **Single agent**: the agent is the only entity acting in the environment
- **Multi-agent**: several agents . . .
  - that *maximize their own performance measure*
  - which is impacted by the actions of the other agents.

Single vs Multi-agent

- **Single agent**: the agent is the only entity acting in the environment
- **Multi-agent**: several agents . . .
  - that *maximize their own performance measure*
  - which is impacted by the actions of the other agents.

Multi-agent: Competitive vs Cooperative

- **Competitive**: agents are in competition
  - zero-sum games (chess, poker)
  - limited resources (competition for food, . . .)
- **Cooperative**: agents are working together (e.g. shared performance measure)
  - no gain from another's loss
- **Mixed**: competitive and cooperative aspects

Pinot & Bardet @ Tour de France 2015, 14ème étape

Deterministic vs Non-deterministic

- **Deterministic**: the next state of the environment is completely determined by the current state and the action executed by the agent.
- **Non-deterministic**: otherwise
  - **Stochastic**: non-deterministic with explicit probabilities

Deterministic vs Non-deterministic

- Complex environments where it is impossible of keeping track of everything may appear and be modeled as non-deterministic.
  - weather, stock market
  - pinball (newtonian physics, but too complex to predict)
- Determinism may be a simplifying assumption for ignoring unlikely events.
  - bit-flip in a computer

```
*x = 10;
...
if (x == 10) {
  // do something
}
```

Episodic vs Sequential

- **Episodic**: the agent's experience is divided into atomic episodes, each episode consists of the agent perceiving the environment and then taking an action.
  - action selection is based only on the last percept
  - e.g.: image classification task
- **Sequential**: the current decision could impact future decisions
  - requires planning ahead
  - e.g.: chess game, driving a car

Static vs Dynamic

- **Static**: the environment does not change while the agent is deliberating
- **Dynamic**: the environment can change while the agent is deliberating
  - **Semi-dynamic**: the environment does not change with the passage of time but the agent's performance measure does

Discrete vs Continuous

- **Discrete**: a finite number of distinct, clearly defined states
  - e.g. chess board
- **Continuous**: infinite number of states
  - typically due to the presence of real numbers (e.g. temperature, position, . . .)

Example Task environments

Task	Observability	Agents	Deterministic	Episodic	Static	Discrete
Chess						

[1] Even though computer deal with discrete numbers
[2] considering the action is to return the fully solved sudoku

Example Task environments

Task	Observability	Agents	Deterministic	Episodic	Static	Discrete
Chess	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker						

[1]Even though computer deal with discrete numbers
[2]considering the action is to return the fully solved sudoku

Example Task environments

Task	Observability	Agents	Deterministic	Episodic	Static	Discrete
Chess	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partial	Multi	Stochastic	Sequential	Static	Discrete
Car driving						

[1] Even though computer deal with discrete numbers
[2] considering the action is to return the fully solved sudoku

Example Task environments

Task	Observability	Agents	Deterministic	Episodic	Static	Discrete
Chess	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partial	Multi	Stochastic	Sequential	Static	Discrete
Car driving	Partial	Multi	Non-deter	Sequential	Dynamic	Continuous
Image class.						

[1] Even though computer deal with discrete numbers
[2] considering the action is to return the fully solved sudoku

Example Task environments

Task	Observability	Agents	Deterministic	Episodic	Static	Discrete
Chess	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partial	Multi	Stochastic	Sequential	Static	Discrete
Car driving	Partial	Multi	Non-deter	Sequential	Dynamic	Continuous
Image class.	Fully	Single	Deterministic	Episodic	Semi	Continuous[1]
Sudoku						

[1] Even though computer deal with discrete numbers
[2] considering the action is to return the fully solved sudoku

Example Task environments

Task	Observability	Agents	Deterministic	Episodic	Static	Discrete
Chess	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partial	Multi	Stochastic	Sequential	Static	Discrete
Car driving	Partial	Multi	Non-deter	Sequential	Dynamic	Continuous
Image class.	Fully	Single	Deterministic	Episodic	Semi	Continuous[1]
Sudoku	Fully	Single	Deterministic	Episodic[2]	Static	Discrete

[1] Even though computer deal with discrete numbers
[2] considering the action is to return the fully solved sudoku

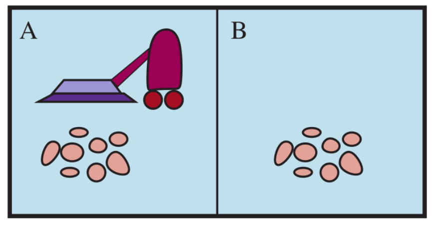
Section 4

Agent structure

Reminder: The Vacuum-cleaner World

A vacuum cleaner world with just two locations, A and B.

- Each location can be either **Dirty** or **Clean**.
- The agent can:
  - move left or right and
  - suck up dirt in the square it occupies.
- The agent perceives:
  - its location,
  - whether there is dirt in the location.



Agent	Percepts	Actions
Cleaner	A, B, Clean, Dirty	Suck, Left, Right

Agent structure: tabulated agent

Fully characterized by the *agent function*, mapping each possible percept sequence to an action.

If we have the tabulated agent function, we can build the agent:

Tabulation of the agent function

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)	Right
...	...

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*

→ *an ideal model, not usable in practice*

Agent structure: a desirable property

The key challenge in AI is to find out how to write programs that can, to the extent possible, produce rational behaviors from a smallish program rather that from a vast table

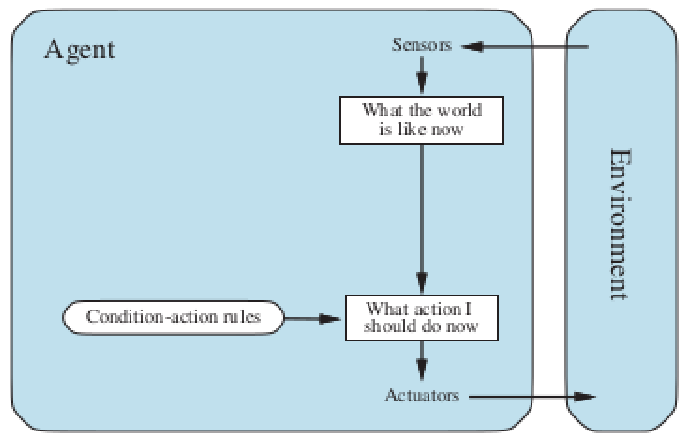
The agent below is equivalent to the table-driven agent, but it is much more compact.

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*

Agent structure: Simple reflex agent

- **Simple reflex agent**: based on *condition-action rules*, applied to the *last percept*.



Legend:

- *rectangle*: current internal state of decision process
- *oval*: background knowledge used in the process

Agent structure: Simple reflex agent

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

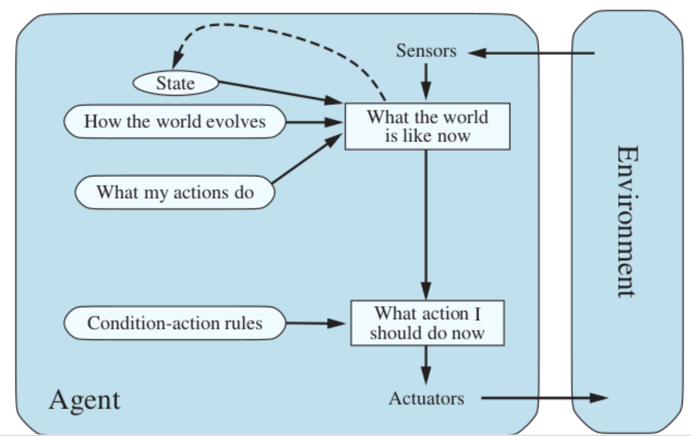
  state ← INTERPRET-INPUT(*percept*)
  rule ← RULE-MATCH(*state*, *rules*)
  action ← *rule*.ACTION
  return *action*

Key limitations:

- partially observable environments (no memory)
- no handling of dynamic environments

Agent structure: Model-based reflex agent

- **Model-based reflex agent**:
  - maintains an *internal state* that depends on the percept history.
  - apply *condition-action rules* to the *internal state*.



Agent structure: Model-based reflex agent

function MODEL-BASED-REFLEX-AGENT(*percept*) **returns** an action
  persistent: *state*, the agent's current conception of the world state
    transition_model, a description of how the next state depends on the current state and 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 ← UPDATE-STATE(*state*, *action*, *percept*, *transition_model*, *sensor_model*)
  rule ← RULE-MATCH(*state*, *rules*)
  action ← *rule*.ACTION
  return *action*

Keeps track of the current state of the world, and choose the next action based on the current state and a set of fixed rules.

Agent structure: Model-based reflex agent

Benefits:

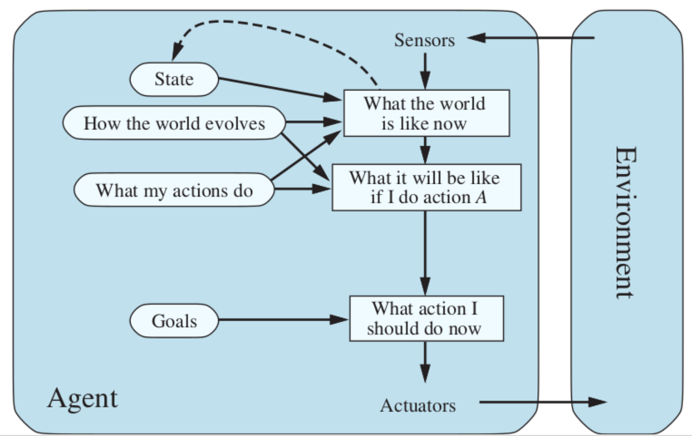
- can handle partially observable environments
- can handle dynamic environments

Limitations:

- very specialized, achieve what is implied by the rules
- rules must cover all plausible situations

Agent structure: Goal-based agent

- **Goal-based agent**:
  - maintains a *goal* that describes situations that are desirable.
  - select action that get it closer to the *goal*.



Agent structure: Goal-based agent

- Goals describe desirable situations (*happy* states)
- The agent selects actions that lead to the goal
  - Often the first step of a long journey
  - enabled by *search* (next course) and *planning* algorithms

Agent structure: Goal-based agent

- Goals describe desirable situations (*happy* states)
- The agent selects actions that lead to the goal
  - Often the first step of a long journey
  - enabled by *search* (next course) and *planning* algorithms

Benefits:

- handling of complex objectives requiring multiple actions
- flexibility (behavior can be changed by changing the goal)

Agent structure: Goal-based agent

- Goals describe desirable situations (*happy* states)
- The agent selects actions that lead to the goal
  - Often the first step of a long journey
  - enabled by *search* (next course) and *planning* algorithms

Benefits:

- handling of complex objectives requiring multiple actions
- flexibility (behavior can be changed by changing the goal)

Limitations:

- crude characterization of the objective (boolean happy/unhappy)
  - what would be the goal of a trading agent?
  - is a destination sufficient to capture the objectives of a self-driving car?

Example: Double or Nothing

Setup: You are in a television game show, and have won 50000 euros so far. You have the choice between:

- stop the game and keep the money
- continue the game, for one last question:
  - if you win, you double the money,
  - otherwise you lose everything.

You estimate your chances of winning the last question to 60%, what would you do?

Example: Double or Nothing

Setup: You are in a television game show, and have won 50000 euros so far. You have the choice between:

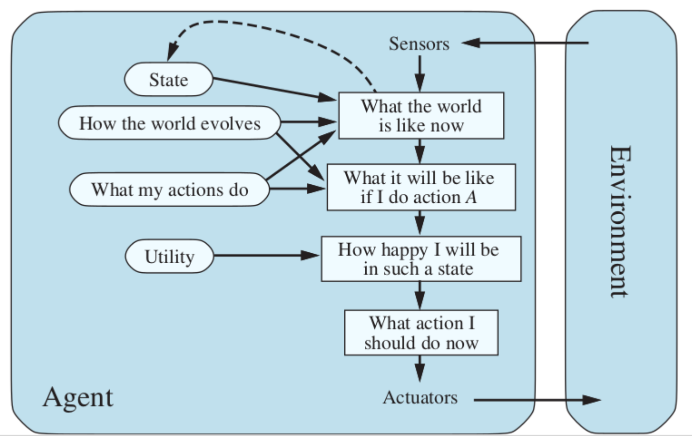
- stop the game and keep the money
- continue the game, for one last question:
  - if you win, you double the money,
  - otherwise you lose everything.

You estimate your chances of winning the last question to 60%, what would you do?

- depends on your utility function:
  - "how happy would you be with 100000 euros?, 50000?, 0?"
- depends on your probability assessment
  - here 0.6 probability of winning

Agent structure: Utility-based agent

- **Utility-based agent**:
  - maintains a *utility function* that assigns a *utility* value to each state.
  - selects actions that maximizes the *expected utility*



Agent structure: Utility-based agent

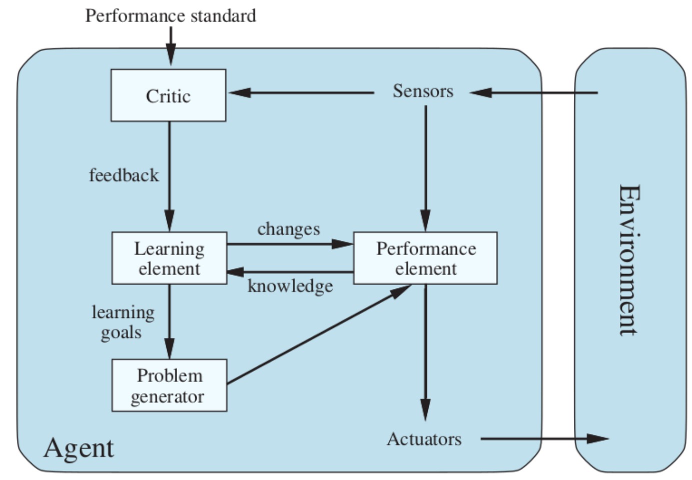
- utility allows capturing complex objectives with trade-offs, differents degrees of satisfaction, etc.
- expected utility deals with uncertain outcomes
- utility should be aligned with the performance measure

Learning agents

Turing (1950):

- actually programming an intelligent agent seems untractable
- proposes to build learning machines and then teach them
- **Learning agent**:
  - improves its performance based on experience.
  - can be any of the previous types of agents.

Learning agents



Learning agents

- **Learning element**: responsible for making improvements.
  - improve the world/action model
  - improve the utility function
  - improve the condition-action rules (reflex agents)
- **Performance element**: responsible for selecting actions.
  - rule-based / search-based
- **Critic**: provides feedback on the agent's behavior.
  - reward signal on desirable behavior
- **Problem generator**: suggests actions that lead to new experiences.
  - improves long time performance by suggesting actions that lead to new experiences (even if they may be bad in the short term)

Section 5

Summary

Summary

- **agent** perceives its environment and acts on it
- **agent function** maps percepts sequences to actions
- **performance measure** evaluates the success of the agent
  - **rational agent** selects actions that maximize the expected performance measure
- **environment** is characterized along several dimensions
  - **fully observable** vs **partially observable**
  - **deterministic** vs **stochastic**
  - **episodic** vs **sequential**
  - **static** vs **dynamic**
  - **discrete** vs **continuous**
  - **single-agent** vs **multi-agent**
- **agent structure** defines the main functionalities
  - **reflex** agent, with or without an environment **model**
  - **goal-based** and **utility-based** agent
- agents can improve their performance through **learning**