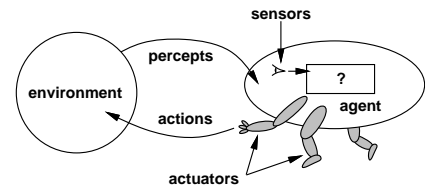


# INTELLIGENT AGENTS

## CHAPTER 2

### Agents and environments



Agents include humans, robots, softbots, thermostats, etc.

The **agent function** maps from percept histories to actions:

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

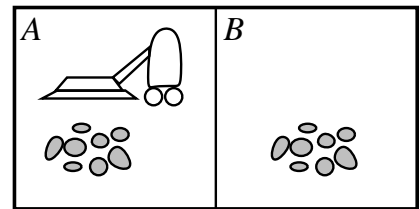
The **agent program** runs on the physical **architecture** to produce  $f$

### Reminders

Assignment 0 (lisp refresher) due 1/28

Lisp/emacs/AIMA tutorial: 11-1 today and Monday, 271 Soda

### Vacuum-cleaner world



Percepts: location and contents, e.g.,  $[A, Dirty]$

Actions: *Left, Right, Suck, NoOp*

### Outline

- ◇ Agents and environments
- ◇ Rationality
- ◇ PEAS (Performance measure, Environment, Actuators, Sensors)
- ◇ Environment types
- ◇ Agent types

### A vacuum-cleaner agent

Percept sequence	Action
$[A, Clean]$	<i>Right</i>
$[A, Dirty]$	<i>Suck</i>
$[B, Clean]$	<i>Left</i>
$[B, Dirty]$	<i>Suck</i>
$[A, Clean], [A, Clean]$	<i>Right</i>
$[A, Clean], [A, Dirty]$	<i>Suck</i>
⋮	⋮

```
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
```

What is the **right** function?

Can it be implemented in a small agent program?

## Rationality

Fixed **performance measure** evaluates the **environment sequence**

- one point per square cleaned up in time  $T$ ?
- one point per clean square per time step, minus one per move?
- penalize for  $> k$  dirty squares?

A **rational agent** chooses whichever action maximizes the **expected** value of the performance measure **given the percept sequence to date**

Rational  $\neq$  omniscient

- percepts may not supply all relevant information

Rational  $\neq$  clairvoyant

- action outcomes may not be as expected

Hence, rational  $\neq$  successful

Rational  $\Rightarrow$  exploration, learning, autonomy

## Internet shopping agent

**Performance measure??**

**Environment??**

**Actuators??**

**Sensors??**

## PEAS

To design a rational agent, we must specify the **task environment**

Consider, e.g., the task of designing an automated taxi:

**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)

## PEAS

To design a rational agent, we must specify the **task environment**

Consider, e.g., the task of designing an automated taxi:

**Performance measure??** safety, destination, profits, legality, comfort, ...

**Environment??** US streets/freeways, traffic, pedestrians, weather, ...

**Actuators??** steering, accelerator, brake, horn, speaker/display, ...

**Sensors??** video, accelerometers, gauges, engine sensors, keyboard, GPS, ...

## Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
<b>Observable??</b>				
<b>Deterministic??</b>				
<b>Episodic??</b>				
<b>Static??</b>				
<b>Discrete??</b>				
<b>Single-agent??</b>				

**Environment types**

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	Yes	Yes	No	No
Deterministic??				
Episodic??				
Static??				
Discrete??				
Single-agent??				

**Environment types**

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	Yes	Yes	No	No
Deterministic??	Yes	No	Partly	No
Episodic??	No	No	No	No
Static??	Yes	Semi	Semi	No
Discrete??				
Single-agent??				

**Environment types**

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	Yes	Yes	No	No
Deterministic??	Yes	No	Partly	No
Episodic??				
Static??				
Discrete??				
Single-agent??				

**Environment types**

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	Yes	Yes	No	No
Deterministic??	Yes	No	Partly	No
Episodic??	No	No	No	No
Static??	Yes	Semi	Semi	No
Discrete??	Yes	Yes	Yes	No
Single-agent??				

**Environment types**

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	Yes	Yes	No	No
Deterministic??	Yes	No	Partly	No
Episodic??	No	No	No	No
Static??				
Discrete??				
Single-agent??				

**Environment types**

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	Yes	Yes	No	No
Deterministic??	Yes	No	Partly	No
Episodic??	No	No	No	No
Static??	Yes	Semi	Semi	No
Discrete??	Yes	Yes	Yes	No
Single-agent??	Yes	No	Yes (except auctions)	No

**The environment type largely determines the agent design**

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

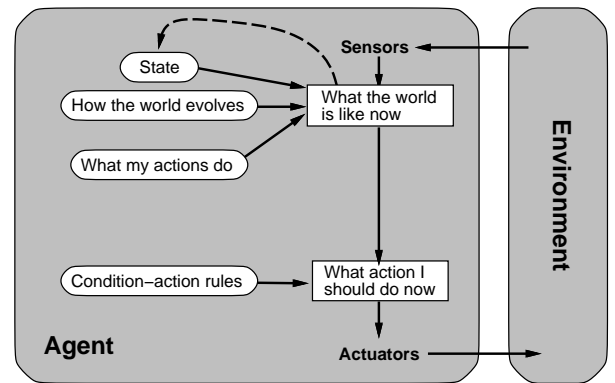
## Agent types

Four basic types in order of increasing generality:

- simple reflex agents
- reflex agents with state
- goal-based agents
- utility-based agents

All these can be turned into learning agents

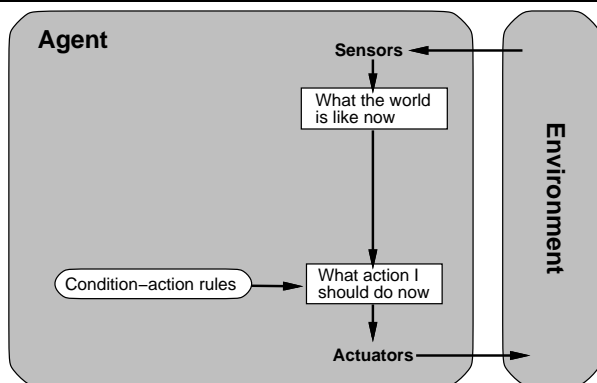
## Reflex agents with state



Chapter 2 19

Chapter 2 22

## Simple reflex agents



Chapter 2 20

## Example

```
function REFLEX-VACUUM-AGENT([location,status]) returns an action
static: last_A, last_B, numbers, initially ∞
    if status = Dirty then ...
```

```
(defun make-reflex-vacuum-agent-with-state-program ()
  (let ((last-A infinity) (last-B infinity))
    #'(lambda (percept)
      (let ((location (first percept)) (status (second percept)))
        (incf last-A) (incf last-B)
        (cond
         ((eq status 'dirty)
          (if (eq location 'A) (setq last-A 0) (setq last-B 0))
          'Suck)
         ((eq location 'A) (if (> last-B 3) 'Right 'NoOp))
         ((eq location 'B) (if (> last-A 3) 'Left 'NoOp))))))))
```

Chapter 2 23

## Example

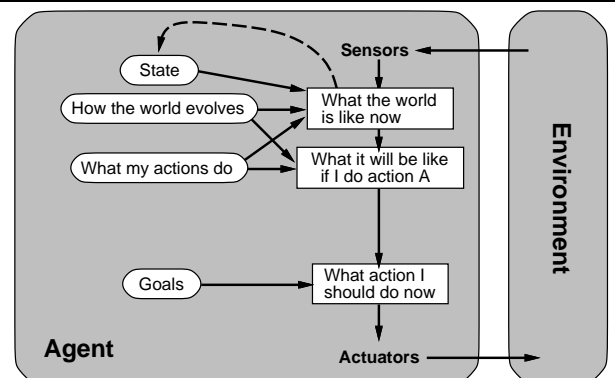
```
function REFLEX-VACUUM-AGENT([location,status]) returns an action
    if status = Dirty then return Suck
    else if location = A then return Right
    else if location = B then return Left
```

```
(setq joe (make-agent :name 'joe :body (make-agent-body)
                     :program (make-reflex-vacuum-agent-program)))
```

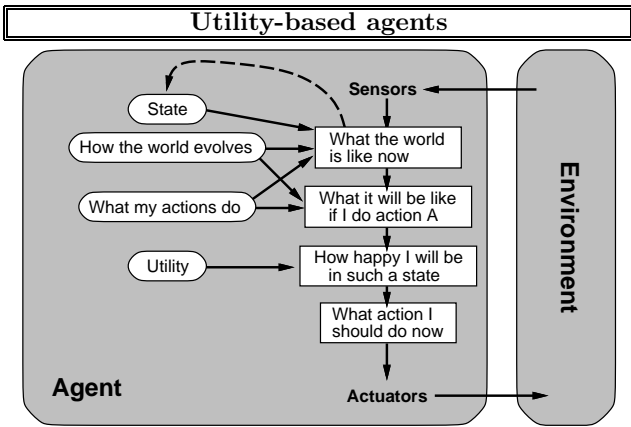
```
(defun make-reflex-vacuum-agent-program ()
  #'(lambda (percept)
    (let ((location (first percept)) (status (second percept)))
      (cond ((eq status 'dirty) 'Suck)
            ((eq location 'A) 'Right)
            ((eq location 'B) 'Left))))))
```

Chapter 2 21

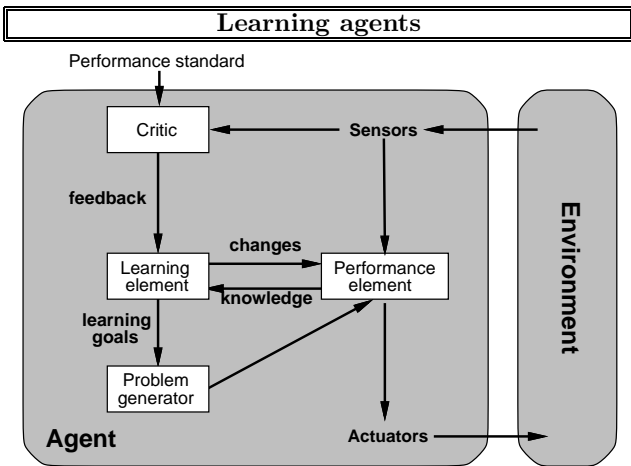
## Goal-based agents



Chapter 2 24



Chapter 2 25



Chapter 2 26

## Summary

Agents interact with environments through actuators and sensors

The agent function describes what the agent does in all circumstances

The performance measure evaluates the environment sequence

A perfectly rational agent maximizes expected performance

Agent programs implement (some) agent functions

PEAS descriptions define task environments

Environments are categorized along several dimensions:

observable? deterministic? episodic? static? discrete? single-agent?

Several basic agent architectures exist:

reflex, reflex with state, goal-based, utility-based

Chapter 2 27