1

Artificial Intelligence
ICS461
Fall 2010

Nancy E. Reed

nreed@hawaii.edu

2

Lecture #2- Intelligent Agents

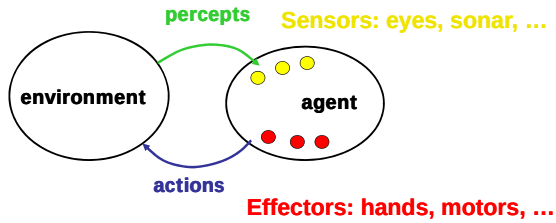
- What is an intelligent agent?
- Agents and environments
- Rationality
- PEAS (Performance measure, Environment, Actuators, Sensors)
- Environment types
- Agent types
- Reference – Ch. 2
- http://aima.cs.berkeley.edu/index.html

3

What is an Autonomous Agent?

Interacts with its environment

1. can sense its environment
2. make decisions, and
3. take action

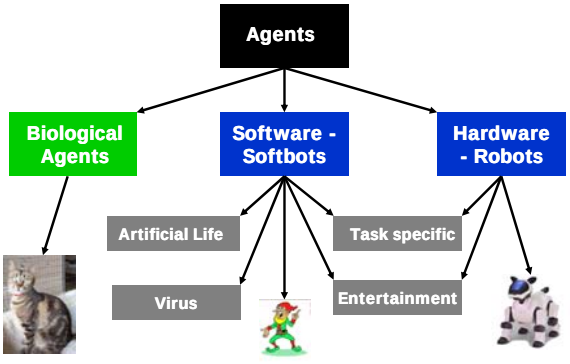


Agents

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

5

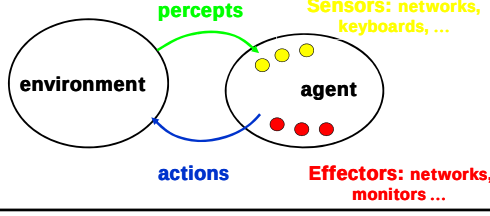
(Physical) Types of Autonomous Agents



6

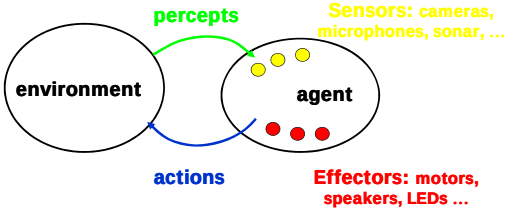
Software Agents - Softbots

- "Live" inside computers and networks
- Percepts: text, images, bits,
- Actions: display information, send messgaes, …
- Examples: simulated pilots, Electric Elves, chat room 'bots', MS "paperclip", …

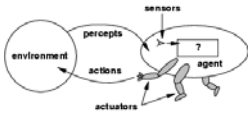


Robotic Agents

- Physical hardware interacts with the environment
- Percepts: images, sound, pressure, acceleration, …
- Actions: movement, sound, light, …

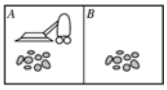


Agents and environments



- 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
- **agent = architecture + program**

Tiny Vacuum-cleaner World



- Percepts: location and contents, e.g., [A,Dirty]
- Actions: *Left*, *Right*, *Suck*, *NoOp*

A vacuum-cleaner agent

- \input{tables/vacuum-agent-function-table}
- http://aima.cs.berkeley.edu/lisp/doc/overview.html

A vacuum-cleaner agent

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*
⋮	⋮

```
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
```

Rational Agents

- An agent should strive to "do the right thing", based on what it can perceive and the actions it can perform.
- The right action is the one that will cause the agent to be most successful
- **Performance measure:** An objective criterion for success of an agent's behavior
- E.g., performance measure of a vacuum-cleaner agent could be amount of dirt cleaned up, amount of time taken, amount of electricity consumed, amount of noise generated, etc.

Rational agents

- **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.

Rational agents

- Rationality is distinct from omniscience (all-knowing with infinite knowledge)
- Agents can perform actions in order to modify future percepts so as to obtain useful information (information gathering, exploration)
- An agent is autonomous if its behavior is determined by its own experience (with ability to learn and adapt)

PEAS Description

- PEAS**: Performance measure, Environment, Actuators, Sensors**
- Must first specify the setting for intelligent agent design

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

1. Performance measure
2. Environment
3. Actuators
4. Sensors

Taxi Example - PEAS

- First specify the setting for agent design
- Consider, e.g., the task of designing an **automated taxi driver:**
- Performance measure: Safe, fast, legal, comfortable trip, maximize profits
- Environment: Roads, other traffic, pedestrians, customers
- Actuators: Steering wheel, accelerator, brake, signal, horn
- Sensors: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard

Medical Example: PEAS

- Agent: Medical diagnosis system
- 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)

Example Robot : PEAS

- Agent: Part-picking robot
- Performance measure: Percentage of parts in correct bins
- Environment: Conveyor belt with parts, bins
- Actuators: Jointed arm and hand
- Sensors: Camera, joint angle sensors

English Tutor Example: PEAS

- Agent: Interactive English tutor
- Performance measure: Maximize student's score on test
- Environment: Set of students
- Actuators: Screen display (exercises, suggestions, corrections)
- Sensors: Keyboard

Environment Characteristics (I)

- Fully observable (vs. partially observable): An agent's sensors give it access to the complete state of the environment at each point in time.
- Deterministic (vs. stochastic): The next state of the environment is completely determined by the current state and the action executed by the agent. (If the environment is deterministic except for the actions of other agents, then the environment is strategic)
- 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.

Environment Characteristics (II)

- Static (vs. dynamic): The environment is unchanged while an agent is deliberating. (The environment is semidynamic if the environment itself does not change with the passage of time but the agent's performance score does)
- Discrete (vs. continuous): A limited number of distinct, clearly defined percepts and actions.
- Single agent (vs. multiagent): An agent operating by itself in an environment.

Environment Type Examples

	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

23

Quiz

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

24

Quiz

	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

Agent functions and programs

- An agent is completely specified by the **agent function** mapping percept sequences to actions
- One agent function (or a small equivalence class) is **rational**
- Aim: find a way to implement the rational agent function concisely

Table-lookup agent

- \input{algorithms/table-agent-algorithm}
- Drawbacks:
  - Huge table
  - Take a long time to build the table
  - No autonomy
  - Even with learning, need a long time to learn the table entries

27

A vacuum-cleaner agent

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

```
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
```

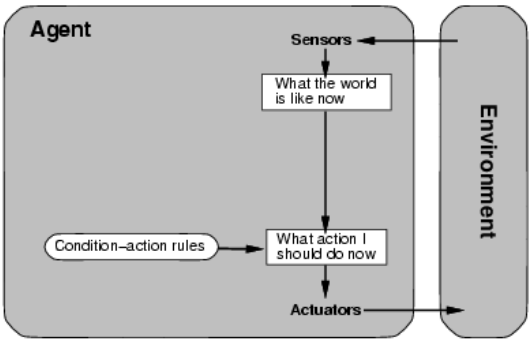
Agent program for a vacuum-cleaner agent

- \input{algorithms/reflex-vacuum-agent-algorithm}

Agent types

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

Simple reflex agents



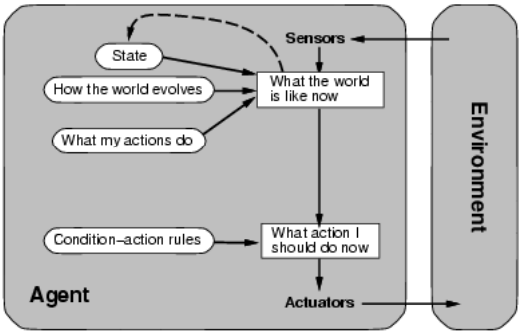
Simple reflex agents

```
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)))))
```

Model-based reflex agents

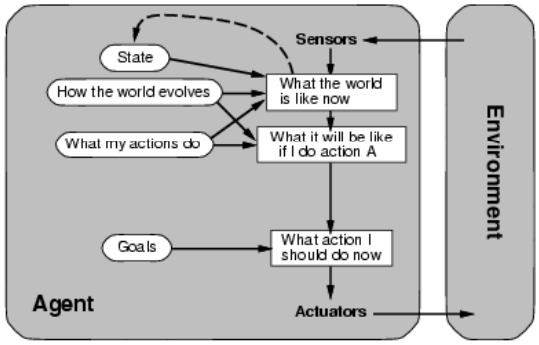


Model-based reflex agents

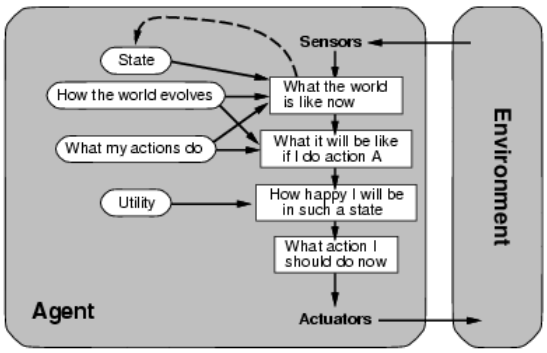
```
function REFLEX-VACUUM-AGENT([location,status]) returns an action
static: last_A, last_B, numbers, initially ∞
  if status = Dirty then ...
```

```
(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)))))))
```

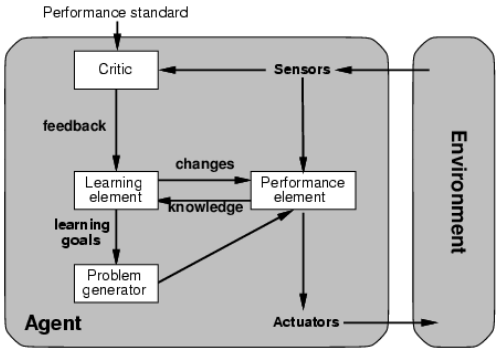
Goal-based agents



Utility-based agents



Learning agents



37

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