MENG 3065 - MODULE 2

Artificial Intelligence: A Modern Approach – Chapter 2 Intelligent Agents

HUMBER

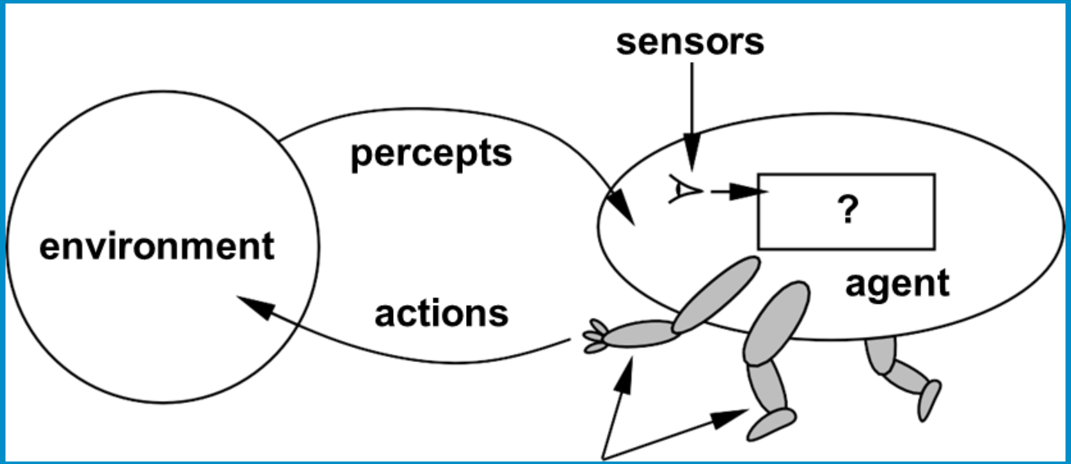
WE ARE HUMBER

Outline

- Intelligent agents (Chapter 2)
  - Intelligent Agents (IA)
  - Environment types
  - IA Behavior
  - IA Structure
  - IA Types

What is an (Intelligent) Agent?

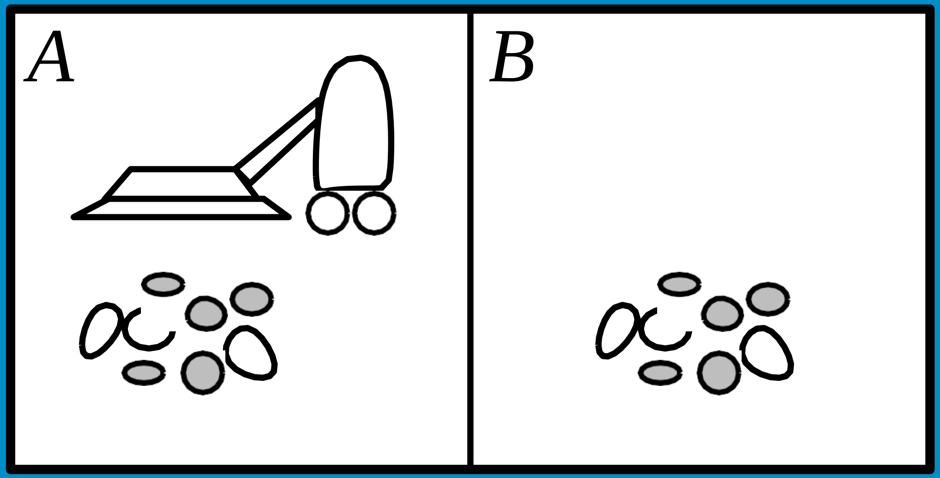
- Agents include humans, robots, softbots, thermostats, etc.
- An agent can be anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
- The agent function maps from percept histories to actions:
  - $f : P^* \rightarrow A$
- The agent program runs on the physical architecture to produce f



What is an (Intelligent) Agent?

- PAGE (Percepts, Actions, Goals, Environment)
- Task-specific & specialized: well-defined goals and environment
- The notion of an agent is meant to be a tool for analyzing systems
  - It is not a different hardware or new programming languages

Vacuum-cleaner world



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

Actions: *Left, Right, Suck, NoOp*

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
```

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 ⇒ exploration, learning, autonomy

PEAS

- To design a rational agent, we must specify the task environment
- Example: 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, engine sensors, keyboard, GPS, . . .

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)

A Windshield Wiper Agent

How do we design a agent that can wipe the windshields when needed?

- Goals? Keep windshields clean & maintain visibility
- Percepts? Raining, Dirty
- Sensors? Camera (moist sensor)
- Effectors? Wipers (left, right, back)
- Actions? Off, Slow, Medium, Fast
- Environment? Inner city, freeways, highways, weather …

Behavior and performance of IAs

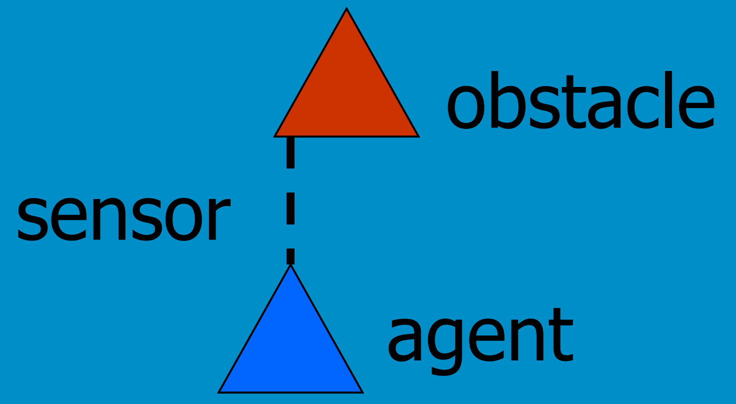
- **Perception** (sequence) to **Action Mapping:** $f: \mathcal{P}^* \rightarrow \mathcal{A}$
  - **Ideal mapping:** specifies which actions an agent ought to take at any point in time
  - **Description:** Look-Up-Table, Closed Form, etc.

- **Performance measure:** a *subjective* measure to characterize how successful an agent is (e.g., speed, power usage, accuracy, money, etc.)

- (degree of) **Autonomy:** to what extent is the agent able to make decisions and take actions on its own?

Look up table

Distance	Action
10	No action
5	Turn left 30 degrees
2	Stop



WE ARE HUMBER

Closed form

- Output (degree of rotation) = F(distance)

- E.g., F(d) = 10/d   (distance cannot be less than 1/10)

How is an Agent different from other software?

- Agents are **autonomous**, that is, they act on behalf of the user
- Agents contain some level of **intelligence**, from fixed rules to learning engines that allow them to adapt to changes in the environment
- Agents don't only act **reactively**, but sometimes also **proactively**

WE ARE HUMBER

How is an Agent different from other software?

- Agents have **social ability**, that is, they communicate with the user, the system, and other agents as required

- Agents may also **cooperate** with other agents to carry out more complex tasks than they themselves can handle

- Agents may **migrate** from one system to another to access remote resources or even to meet other agents

Environment Types

- Characteristics
  - Fully observable vs. partially observable
  - Single agent vs. multiagent
  - Deterministic vs. stochastic
  - Episodic vs. sequential
  - Static vs. dynamic
  - Discrete vs. continuous
  - Others

Environment Types

- Characteristics
  - Fully observable vs. partially observable (Accessible vs. inaccessible)
    - Sensors give access to **complete** state of the environment.
  - Deterministic vs. nondeterministic
    - The next state can be determined based on the current state and the action.
  - Episodic vs. nonepisodic (Sequential)
    - Episode: each perceive and action pairs
    - The quality of action does not depend on the previous episode.

Environment Types

- Characteristics
  - Static vs. dynamic
    - Dynamic if the environment changes during deliberation
  - Discrete vs. continuous
    - The discrete/continuous distinction applies to the **state** of the environment, to the way **time** is handled, and to the **percepts** and **actions** of the agent
    - Chess vs. driving

WE ARE HUMBER

Environment types

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partially	Multi	Stochastic	Sequential	Static	Discrete
Backgammon	Fully	Multi	Stochastic	Sequential	Static	Discrete
Taxi driving	Partially	Multi	Stochastic	Sequential	Dynamic	Continuous
Medical diagnosis	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Image analysis	Fully	Single	Deterministic	Episodic	Semi	Continuous
Part-picking robot	Partially	Single	Stochastic	Episodic	Dynamic	Continuous
Refinery controller	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Interactive English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Figure 2.6 Examples of task environments and their characteristics.

Environment types

Environment	Accessible	Deterministic	Episodic	Static	Discrete
Operating System	Yes	Yes	No	No	Yes
Virtual Reality	Yes	Yes	Yes/no	No	Yes/no
Office Environment	No	No	No	No	No
Mars	No	Semi	No	Semi	No

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

Structure of Intelligent Agents

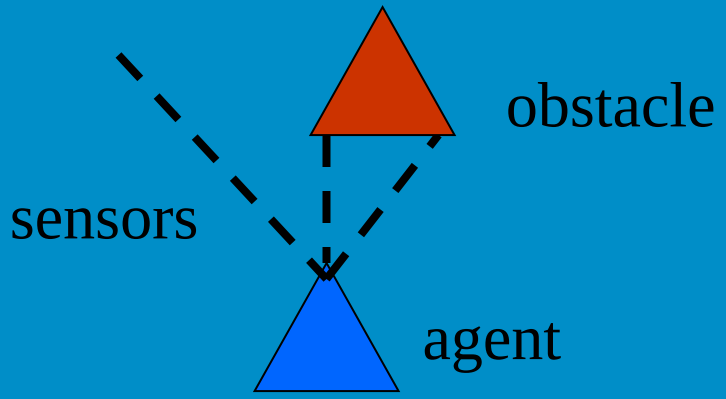
- Agent = architecture + program
- **Agent program:** the implementation of $f: \mathcal{P}^* \rightarrow \mathcal{A}$, the agent's perception-action mapping

```
function Skeleton-Agent(Percept) returns Action
  memory ← UpdateMemory(memory, Percept)
  Action ← ChooseBestAction(memory)
  memory ← UpdateMemory(memory, Action)
return Action
```

- **Architecture:** a device that can execute the agent program (e.g., general-purpose computer, specialized device, robot, etc.)

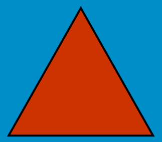
Using a look-up-table to encode $f : \mathcal{P}^* \rightarrow \mathcal{A}$

- **Example:** Collision Avoidance
  - Sensors: 3 proximity sensors
  - Effectors: Steering Wheel, Brakes
- How to generate?
- How large?
- How to select action?



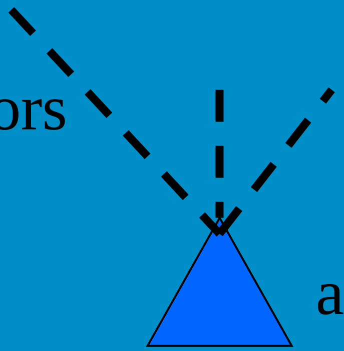
WE ARE HUMBER

Using a look-up-table to encode $f : \mathcal{P}^* \to \mathcal{A}$



- Example: Collision Avoidance
  - Sensors: 3 proximity sensors - possible readings (close, medium, far)
  - Effectors: Steering Wheel - possible actions (left, straight, right), Brakes - possible actions (on or off)
- How to generate: for each $p \in \mathcal{P}_\ell \times \mathcal{P}_m \times \mathcal{P}_r$ generate an appropriate action, $a \in \mathcal{S} \times \mathcal{B}$
- How large:
  - size of table = #possible percepts times # possible actions
    $= |\mathcal{P}_\ell|\ |\mathcal{P}_m|\ |\mathcal{P}_r|\ |\mathcal{S}|\ |\mathcal{B}|$

  E.g., P = {close, medium, far}^{3}
  A = {left, straight, right} $\times$ {on, off}
  then size of table = 27*3*2 = 162



- How to select action? Search.

Agent Classes

- Russell & Norvig (2003) group agents into five classes based on their degree of perceived intelligence and capability

1. Simple reflex agents
2. Model-based reflex agents
3. Goal-based agents
4. Utility-based agents
5. Learning agents

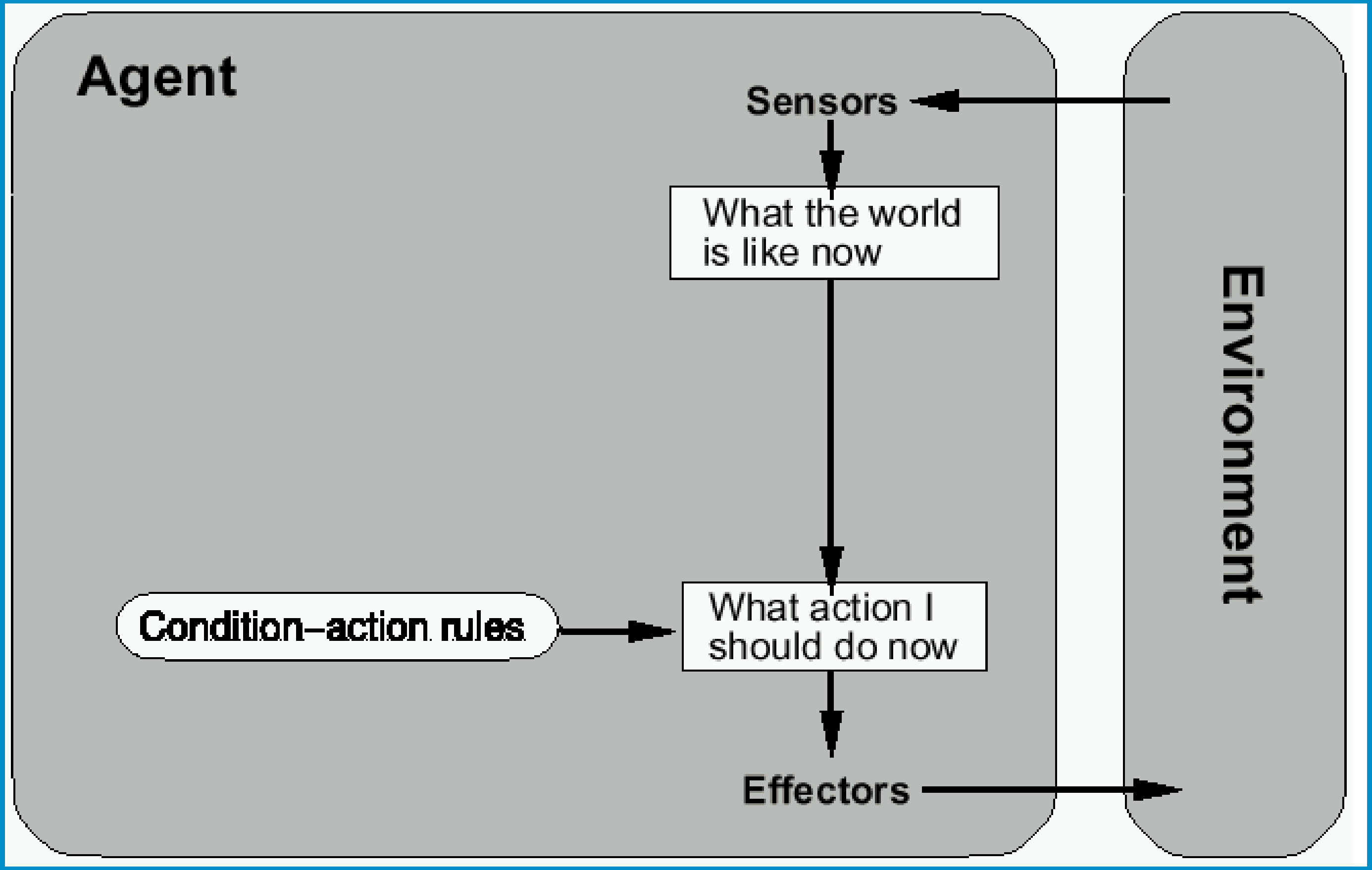
Agent types (Cont'd)

- Reflex agents
  - Reactive: No memory
- Model-based Reflex agents (with internal states)
  - W/o previous state, may not be able to make decision
    - E.g. brake lights at night.
- Goal-based agents
  - Goal information needed to make decision

Agent types (Cont’d)

- Utility-based agents
  - How well can the goal be achieved (degree of happiness)
  - What to do if there are conflicting goals?
    - Speed and safety
  - Which goal should be selected if several can be achieved?

Reflex agents



Example

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

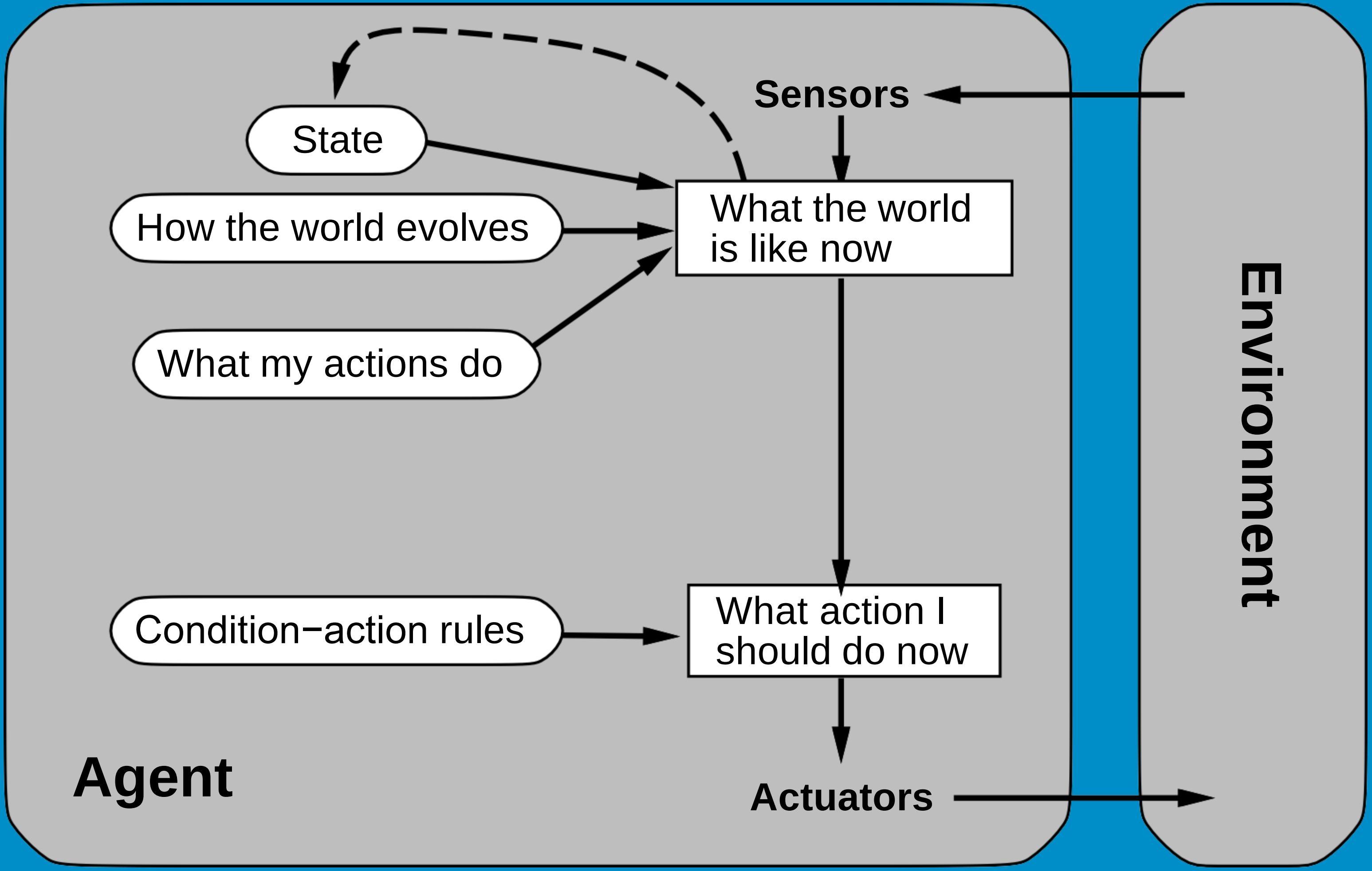
  state ← INTERPRET-INPUT(*percept*)
  rule ← RULE-MATCH(*state*, *rules*)
  action ← *rule*.ACTION
  return *action*

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*

Reactive agents

- Reactive agents do not have internal symbolic models.
- Act by stimulus-response to the current state of the environment.
- Each reactive agent is simple and interacts with others in a basic way.
- Complex patterns of behavior emerge from their interaction.

Model-based Reflex agents (w/ state)



Example

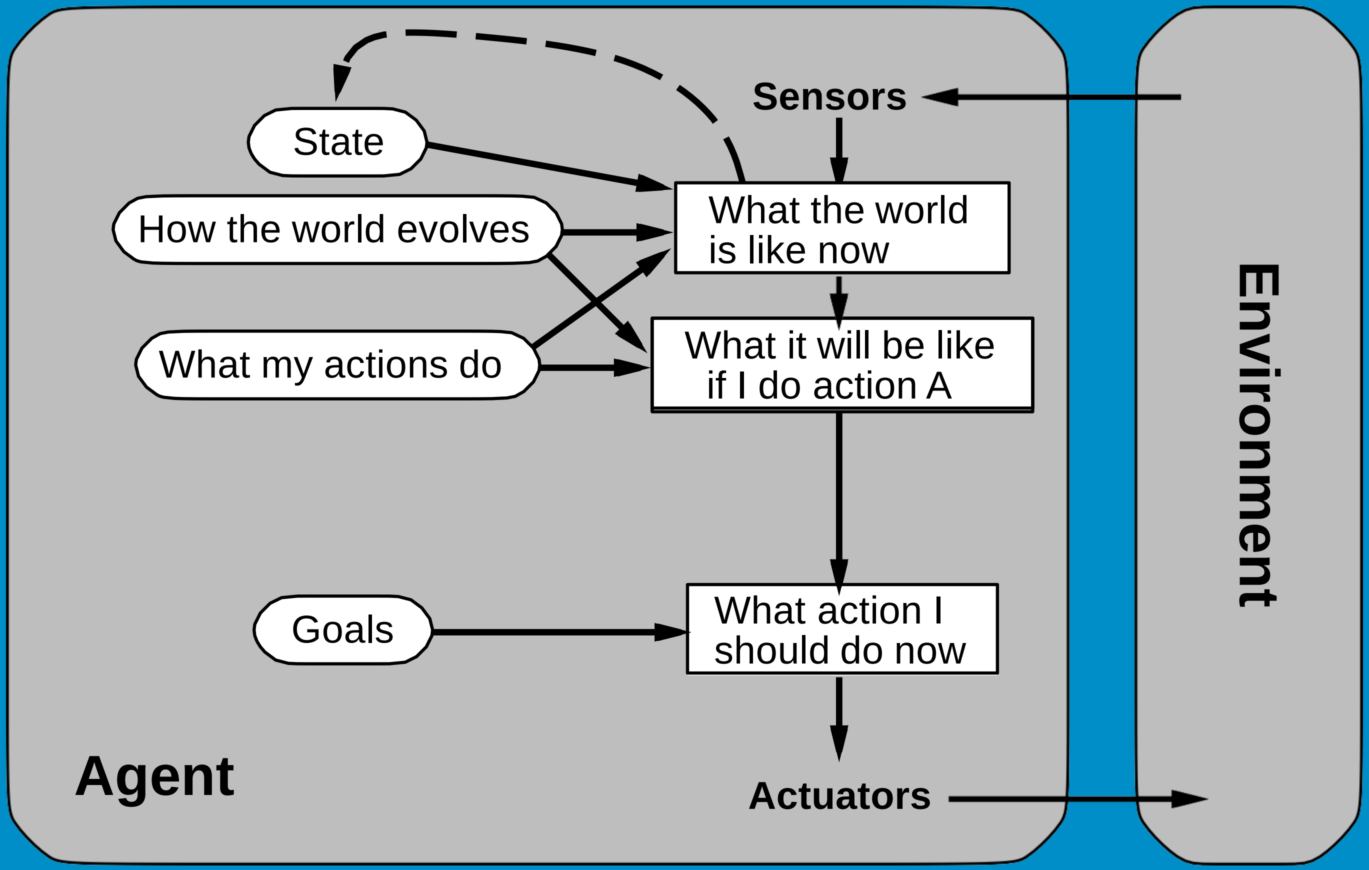
function Reflex-Vacuum-Agent([*location*,*status*]) returns an action
static: *last_A*, *last_B*, numbers, initially ∞

if *status* = *Dirty* then . . .

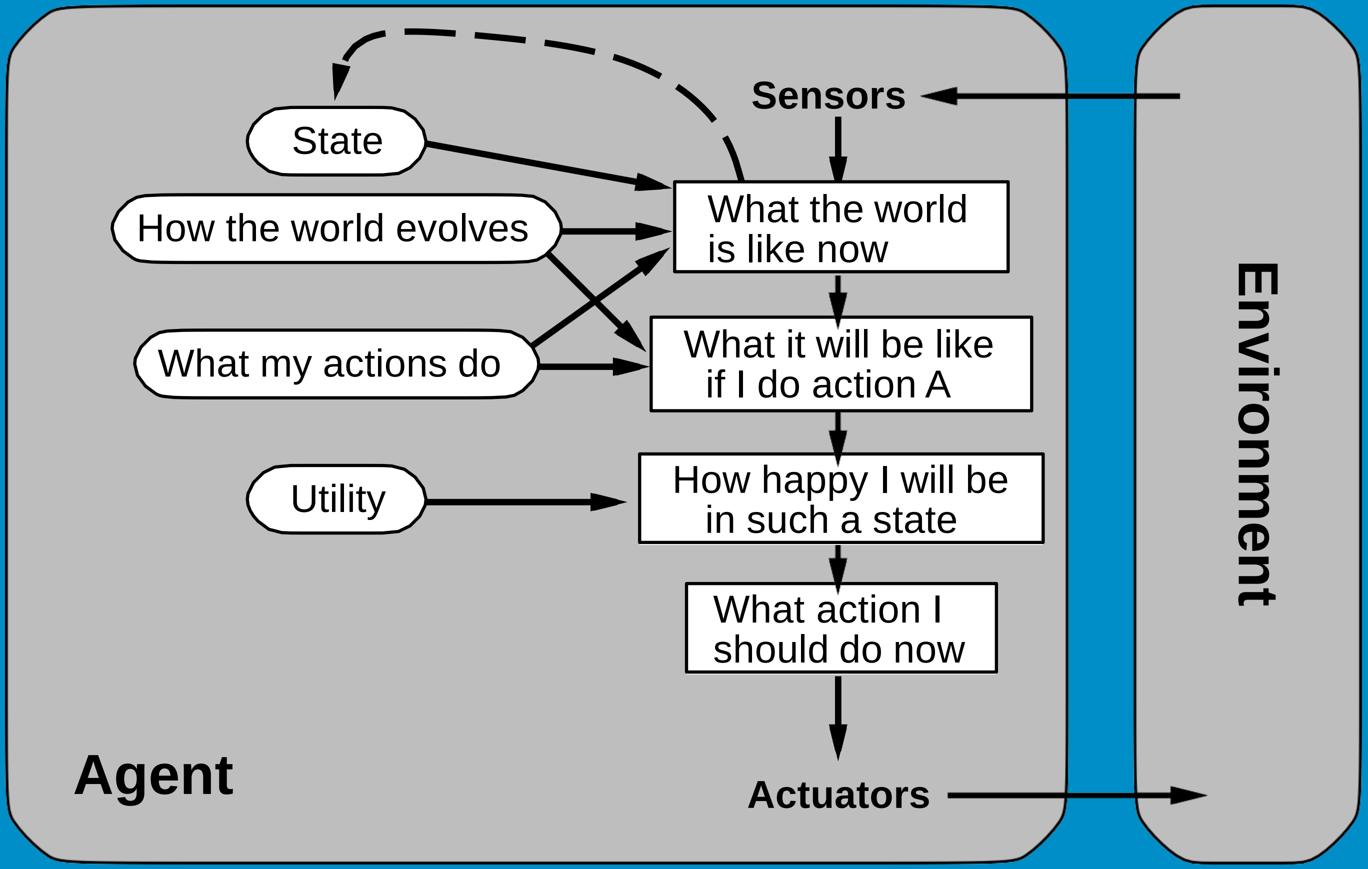
```
(defun make-reflex-vacuum-agent-with-state-program ()
  (let ((last-A infinity) (last-B infinity))
  #'(lambda (percept)
      (let ((location (firstpercept)) (status (second percept)))
        (incf last-A) (incf last-B)
        (cond
         ((eq status 'dirty)
          (if (eq location 'A) (setq 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)))))))
```

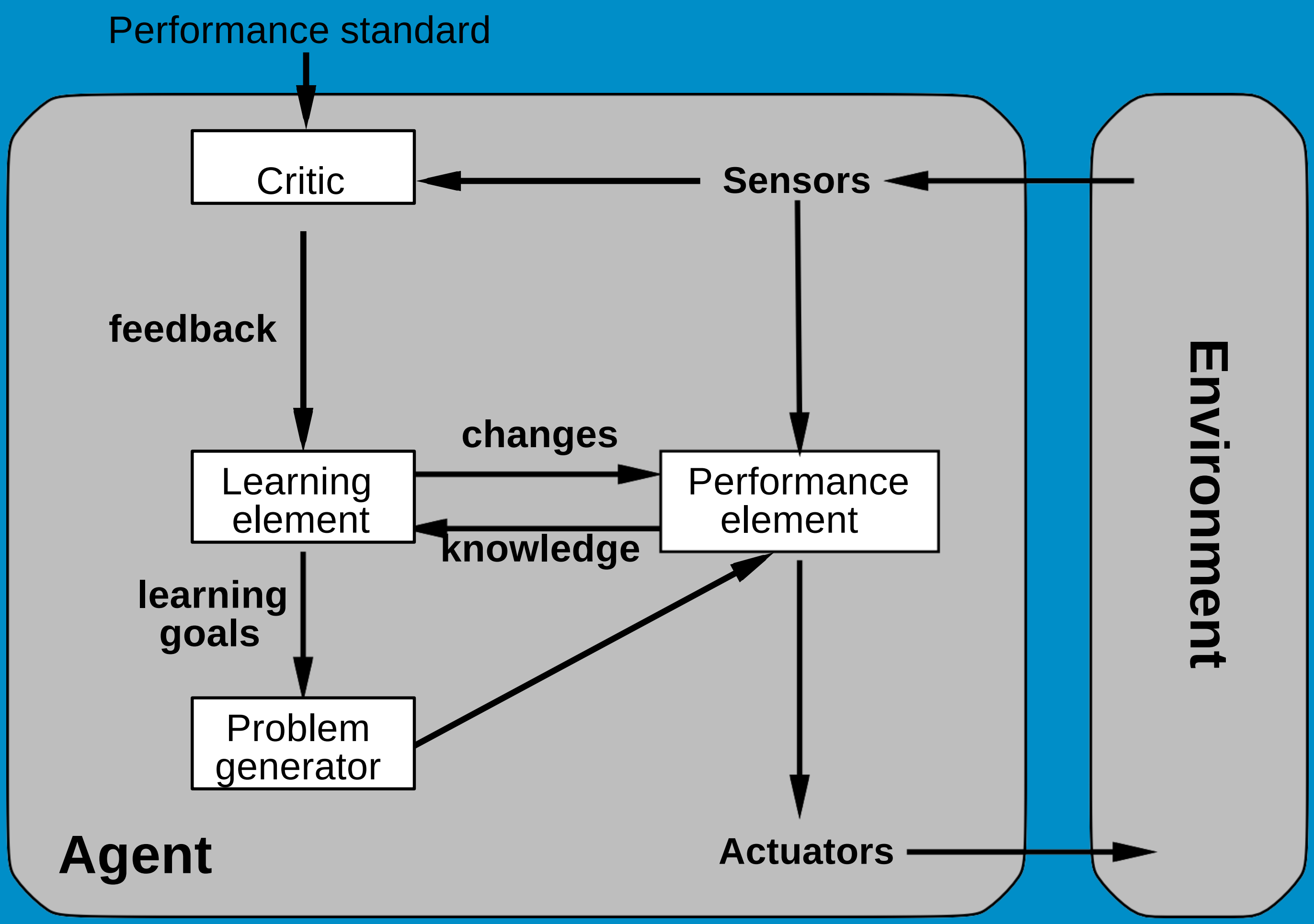
Goal-based agents



Utility-based agents

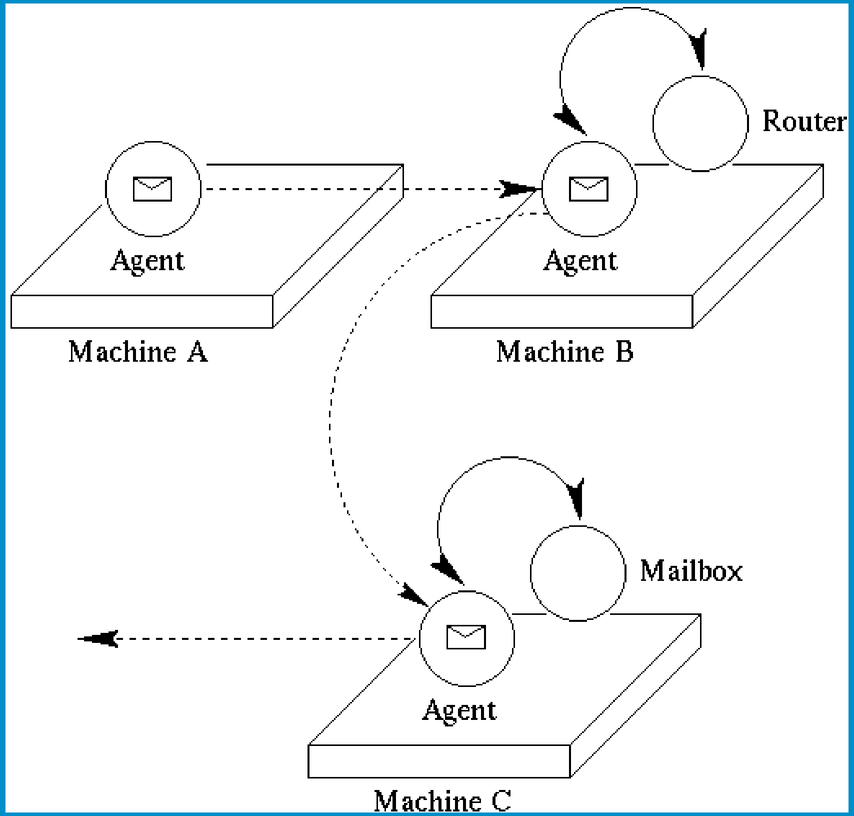


Learning agents



Mobile agents

- Programs that can migrate from one machine to another.
- Execute in a platform-independent execution environment.
- Require agent execution environment (places).
- Mobility not necessary or sufficient condition for agenthood.



A mail agent

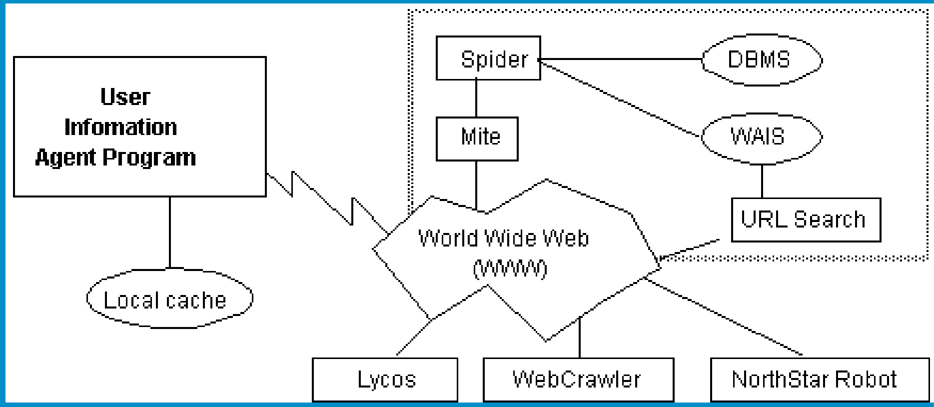
Mobile agents

- Practical but non-functional advantages:
  - Reduced communication cost (e.g. from PDA)
  - Asynchronous computing (when you are not connected)
- Two types:
  - One-hop mobile agents (migrate to one other place)
  - Multi-hop mobile agents (roam the network from place to place)

Mobile agents

- Applications:
  - Distributed information retrieval.
  - Telecommunication network routing.

Information agents



- Manage the explosive growth of information.
- Manipulate or collate information from many distributed sources.
- Information agents can be mobile or static.
- Examples:
  - BargainFinder comparison shops among Internet stores for CDs
  - Internet Softbot infers which internet facilities (finger, ftp, gopher) to use and when from high-level search requests.
- Challenge: ontologies for annotating Web pages (eg, SHOE).

Summary

- **Intelligent Agents:**
  - Anything that can be *viewed as* **perceiving** its **environment** through **sensors** and **acting** upon that environment through its **effectors** to maximize progress towards its **goals**.
  - PAGE (Percepts, Actions, Goals, Environment)
  - Described as a Perception (sequence) to Action Mapping: $f: \mathcal{P}^* \rightarrow \mathcal{A}$
  - Using look-up-table, closed form, etc.

- **Agent Types:** Reflex, Model(state)-based, goal-based, utility-based

- **Rational Action:** The action that maximizes the expected value of the performance measure <u>given the percept sequence to date</u>