Artificial Intelligence

Chapter 2

Stuart RUSSEL

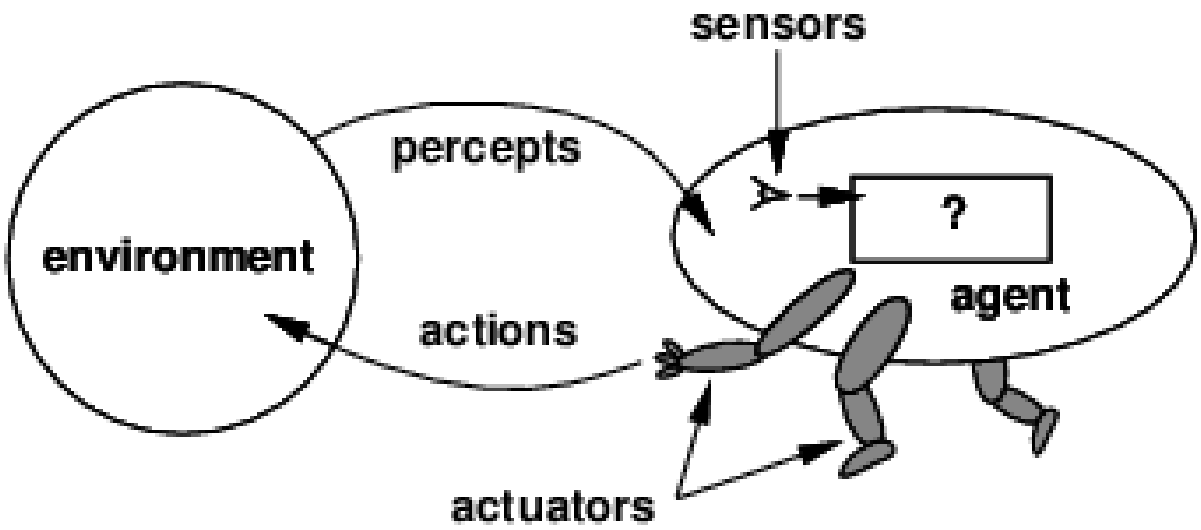
reorganized by L. Aszalós

February 23, 2017

Outline

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

Agents and environments

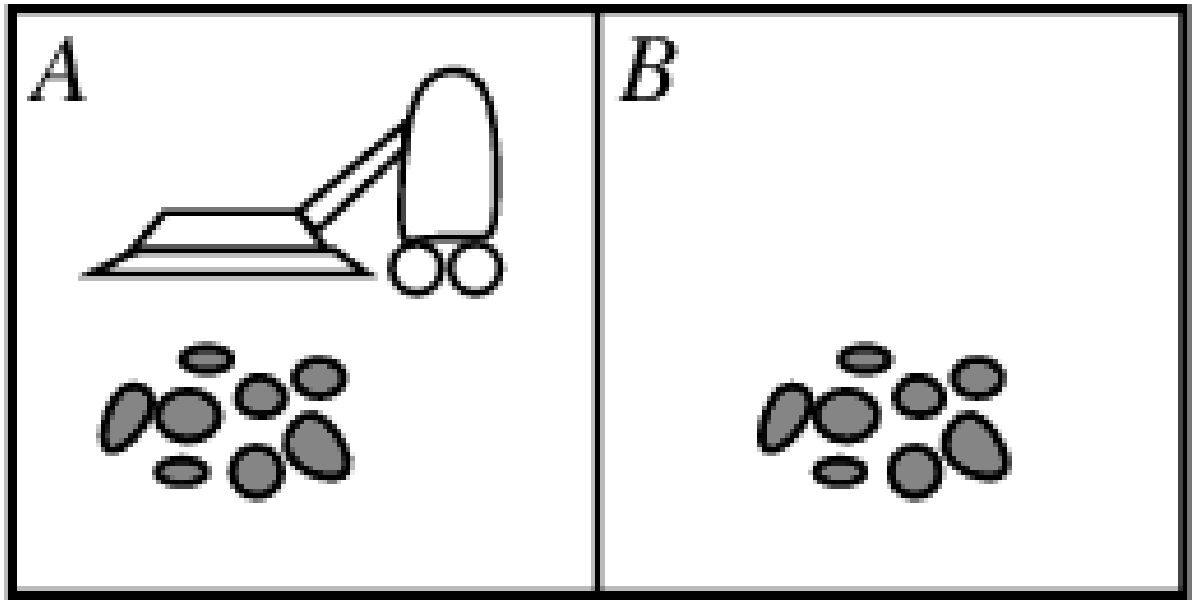


Agents include humans, robots, softbots, thermostats, etc.
The **agent function** maps from percept histories to actions:

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

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

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

```
func Reflex-Vacuum-Agent(location,status): action
    if status == Dirty then return Suck
    else if location == A then return Right
    else if location == B then return Left
```

A vacuum-cleaner agent

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?

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?

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?

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*

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

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

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

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

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

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

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

PEAS

To design a rational agent, we must specify the *task environment*
Consider, e.g., the task of designing an automated taxi:

- *Performance measure*

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, . . .

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*

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, . . .

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*

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, ...

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*

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, ...

Internet shopping agent

- Performance measure
- Environment
- Actuators
- Sensors

Internet shopping agent

- *Performance measure*

Internet shopping agent

- *Performance measure*
  - price, quality, appropriateness, efficiency

Internet shopping agent

- *Performance measure*
  - price, quality, appropriateness, efficiency
- *Environment*

Internet shopping agent

- *Performance measure*
  - price, quality, appropriateness, efficiency
- *Environment*
  - current and future WWW sites, vendors, shippers

Internet shopping agent

- *Performance measure*
  - price, quality, appropriateness, efficiency
- *Environment*
  - current and future WWW sites, vendors, shippers
- *Actuators*

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

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*

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)

Environment types

If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is **fully observable**.

Type	Solitaire	Backgammon	Internet shopping	Taxi
Observable				

Environment types

If the next state of the environment is completely determined by the current state and the action executed by the agent, then we say the environment is **deterministic**.

Type	Solitaire	Backgammon	Internet shopping	Taxi
Observable	yes	yes	no	no
Deterministic				

Environment types

In an episodic task environment, the agent's experience is divided into atomic episodes. In each episode the agent receives a percept and then performs a single action.

Type	Solitaire	Backgammon	Internet shopping	Taxi
Observable	yes	yes	no	no
Deterministic	yes	no	partly	no
Episodic				

Environment types

If the environment can change while an agent is deliberating, then we say the environment is **dynamic** for that agent. (If the environment itself does not change with the passage of time but the agent's performance score does, then we say the environment is **semidynamic**)

Type	Solitaire	Backgammon	Internet shopping	Taxi
Observable	yes	yes	no	no
Deterministic	yes	no	partly	no
Episodic	no	no	no	no
Static				

Environment types

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

Type	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				

Environment types

Type	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

Type	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 (wo 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

All these can be turned into learning agents

Agent types

Four basic types in order of increasing generality:

- simple reflex agents
- reflex agents with state

All these can be turned into learning agents

Agent types

Four basic types in order of increasing generality:

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

All these can be turned into learning agents

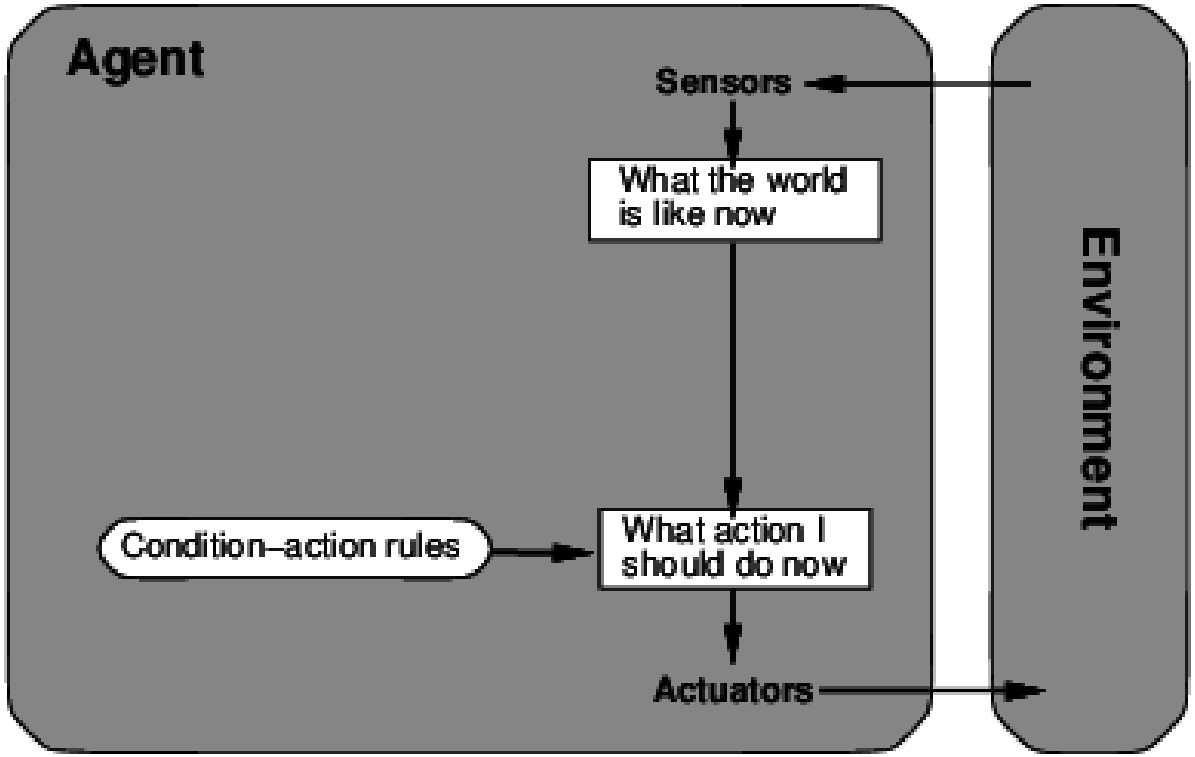
Agent types

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

Simple reflex agents



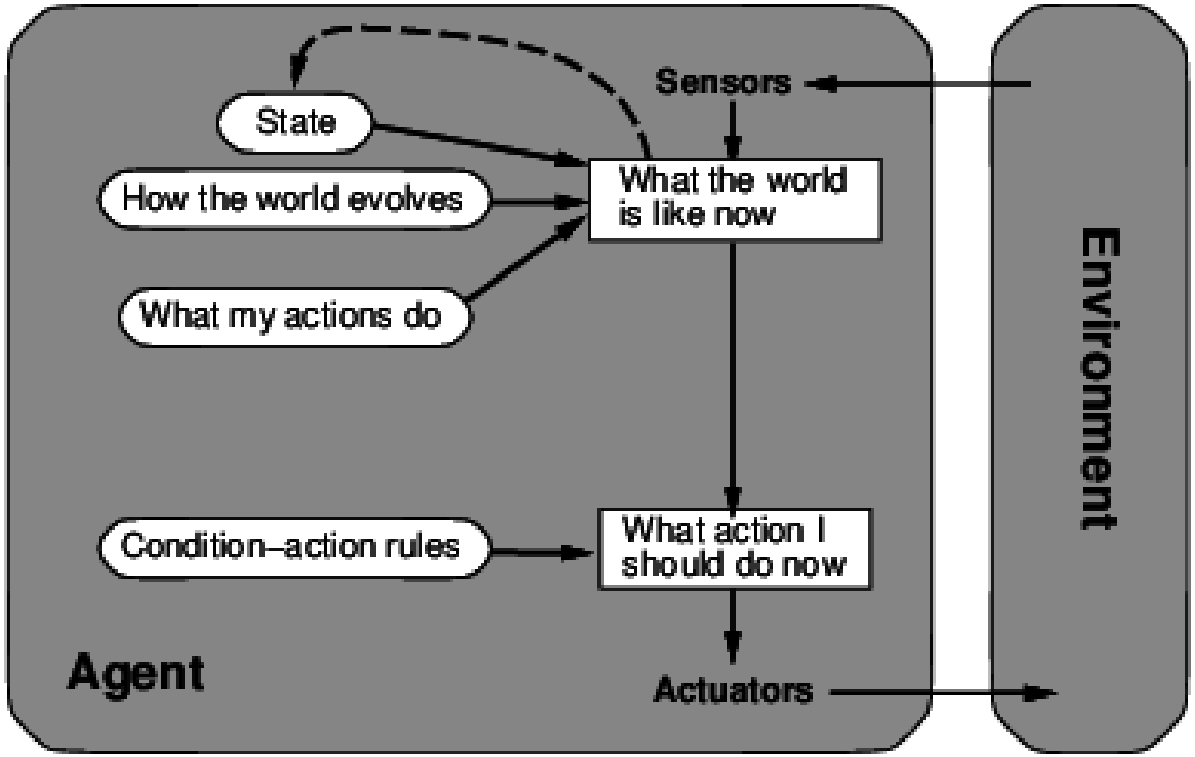
Example

```
# The two locations for the Vacuum world
loc_A, loc_B = (0, 0), (1, 0)

class ReflexVacuumAgent(Agent):
    "A reflex agent for the two-state vacuum environment."

    def __init__(self):
        Agent.__init__(self)
        def program(arg):
            location, status = arg
            if status == 'Dirty': return 'Suck'
            elif location == loc_A: return 'Right'
            elif location == loc_B: return 'Left'
        self.program = program
```

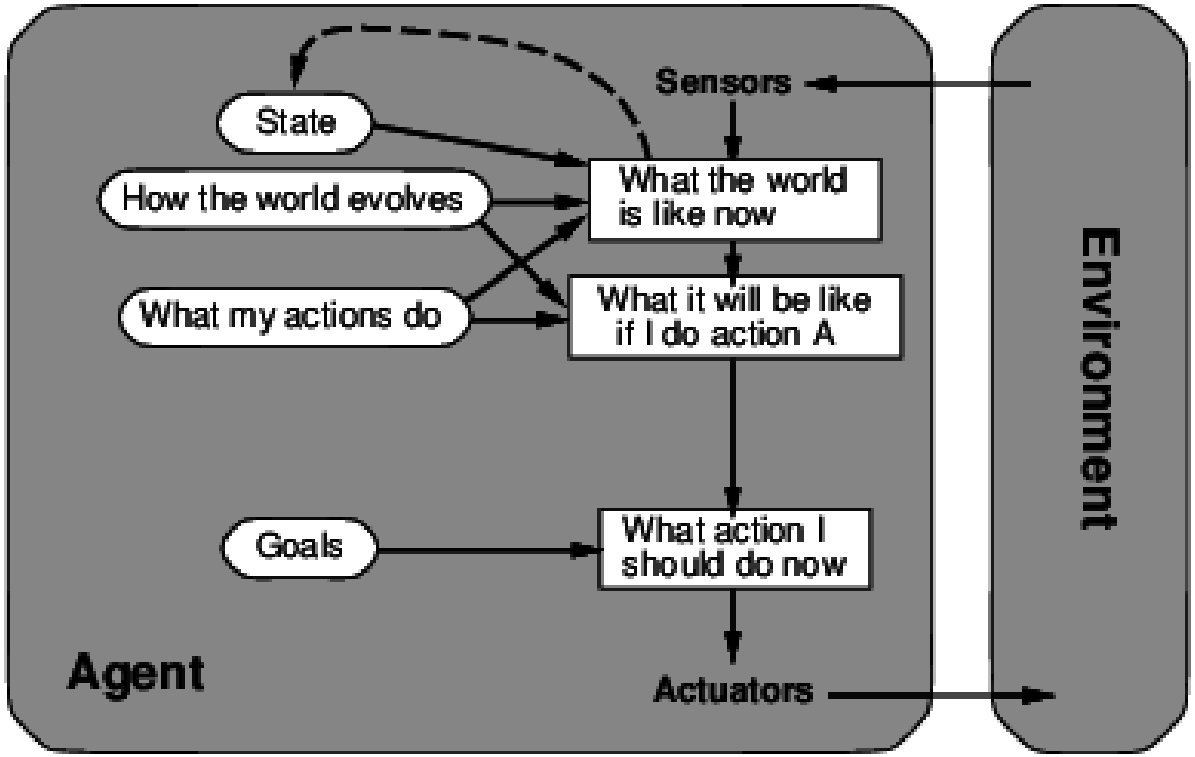
Reflex agents with state



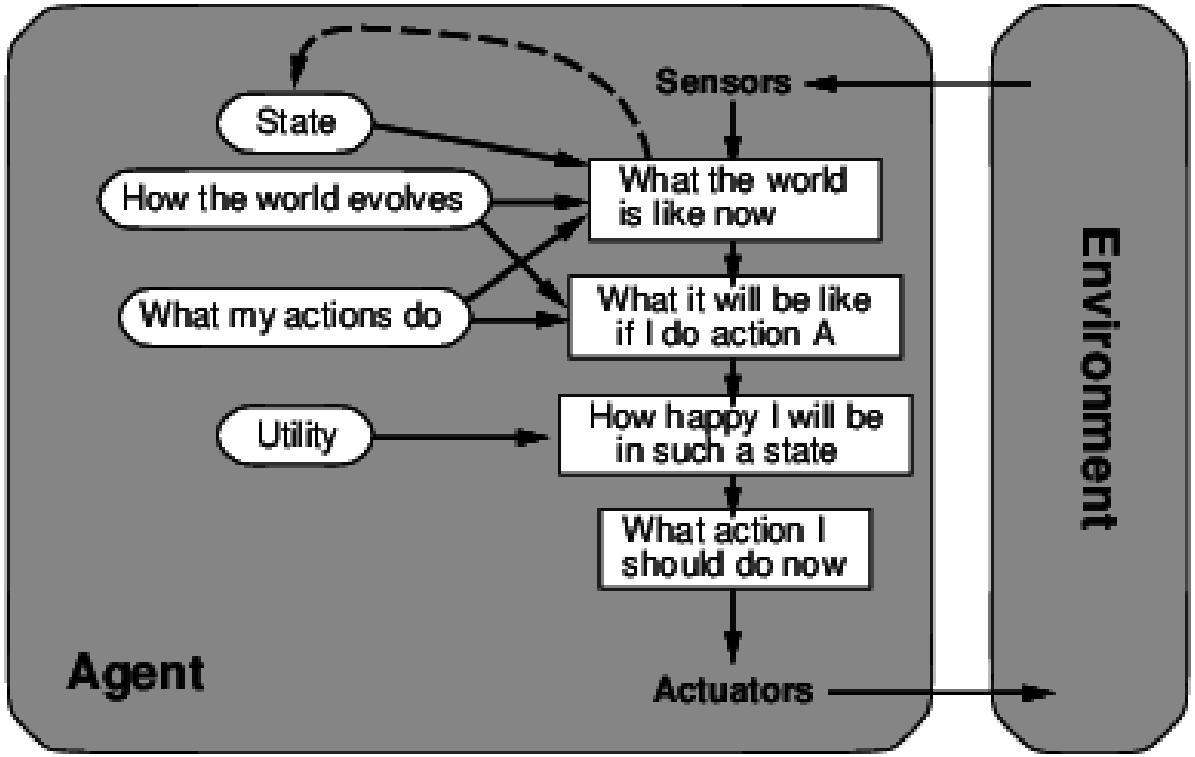
Example

```
class ModelBasedVacuumAgent(Agent):
    "An agent that keeps track of what locations
    are clean or dirty."
    def __init__(self):
        Agent.__init__(self)
        model = {loc_A: None, loc_B: None}
        def program(arg):
            location, status = arg
            model[location] = status ## Update
            if model[loc_A] == model[loc_B] == 'Clean':
                return 'NoOp'
            elif status == 'Dirty': return 'Suck'
            elif location == loc_A: return 'Right'
            elif location == loc_B: return 'Left'
        self.program = program
```

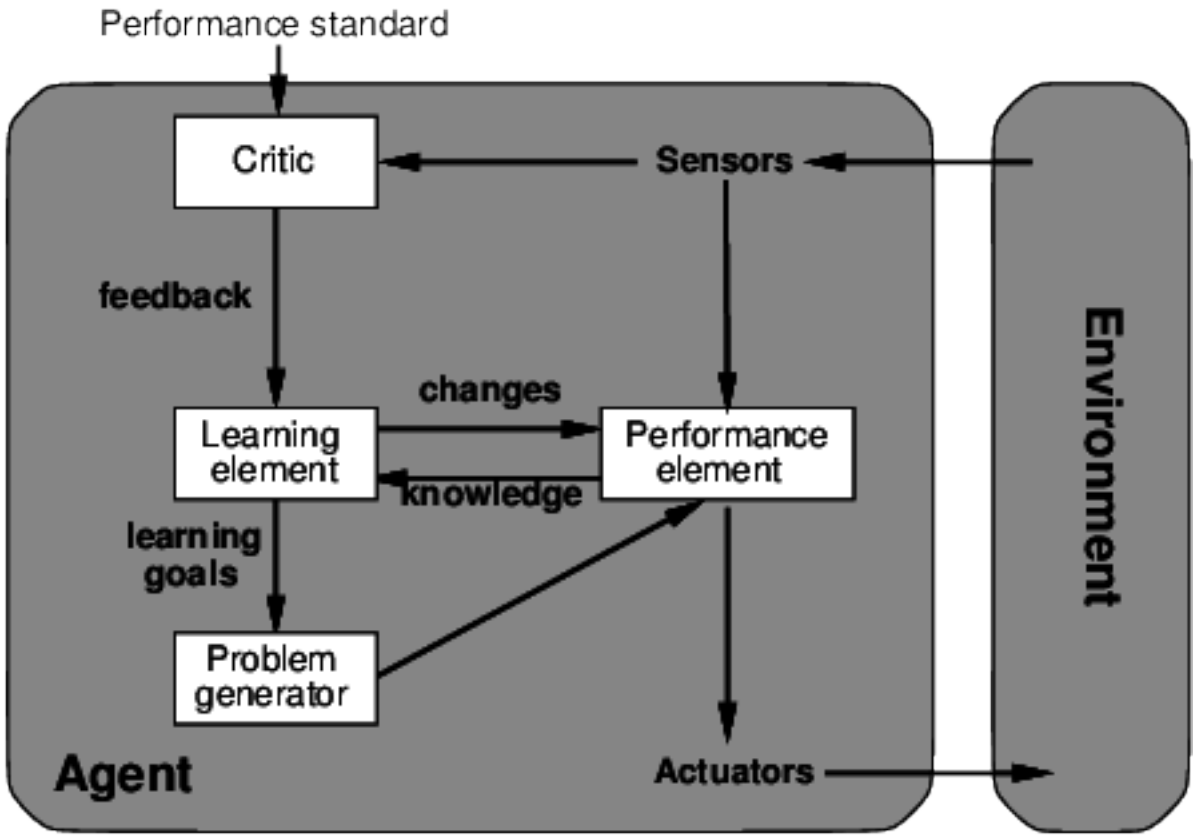
Goal-based agents



Utility-based agents



Learning agents



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**