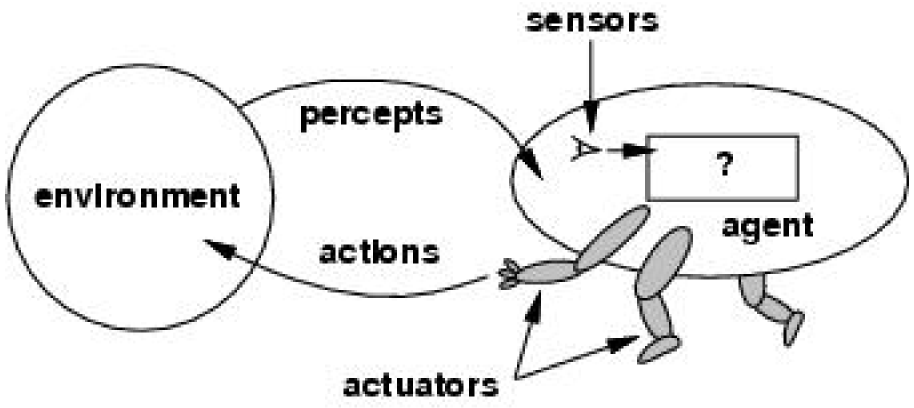
Agents and their behavior

Fundamentals of AI

Learning Outcomes

- Agents and environments
- The concept of rational behavior.
- Environments.
- Agent structure.

Agents and environments

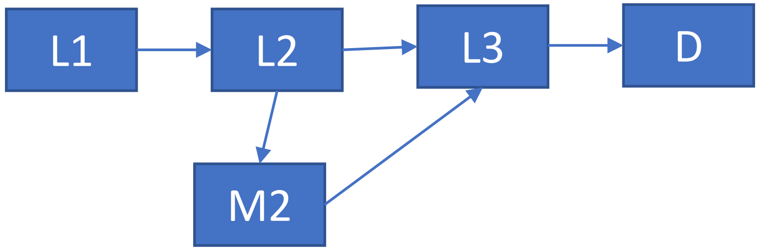


An agent is any entity that takes actions based on inputs that are sensed from the environment.
Agents come in many forms, including: humans, robots, softbots, thermostats, etc.
The *agent function* maps a *percept* sequence to actions

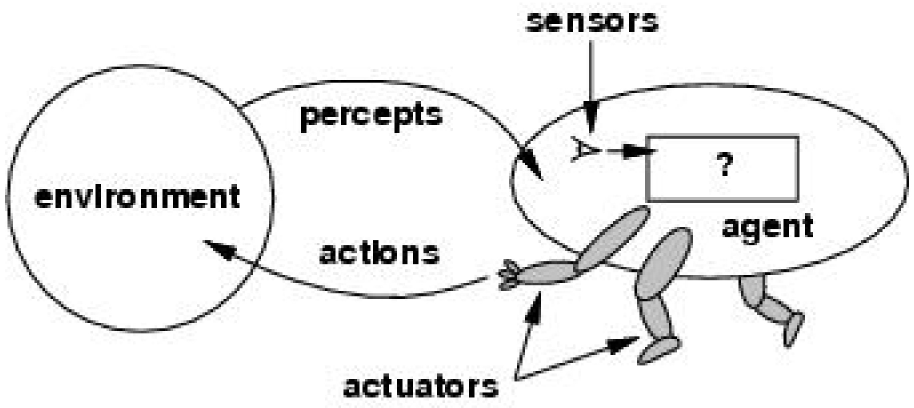
- a thermostat adjusts uses a rheostat to cut off the supply of electricity to a heating element when the temperature rises above a threshold value
- a robot with sufficient memory can use known landmark points in its path to decide its next move

L1, L2, L3-→ D

L1, L2, → M2

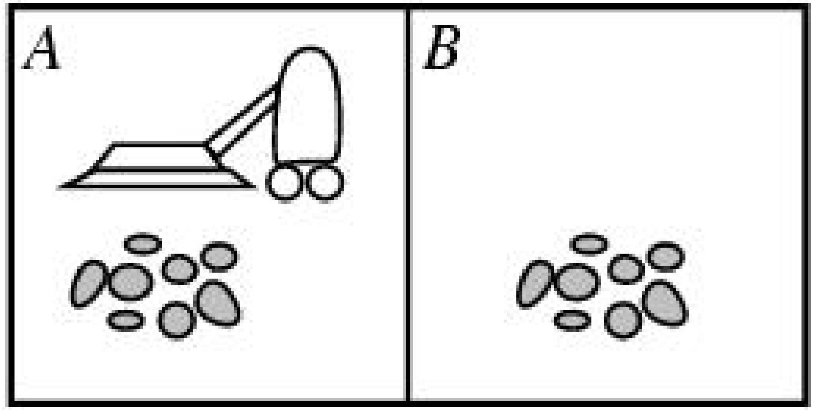


Agents and environments



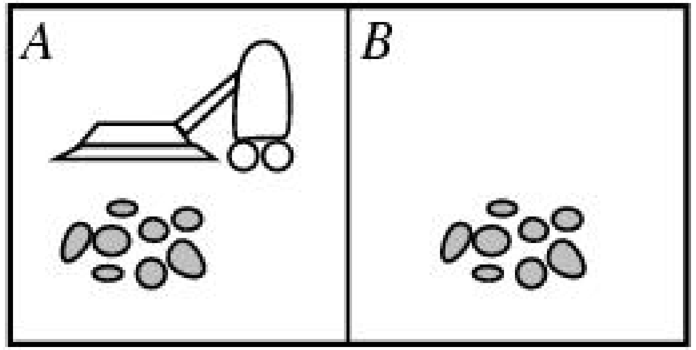
- The *agent functionality* is implemented by a program.
- The agent program interfaces to the actuators to perform actions

The vacuum-cleaner world



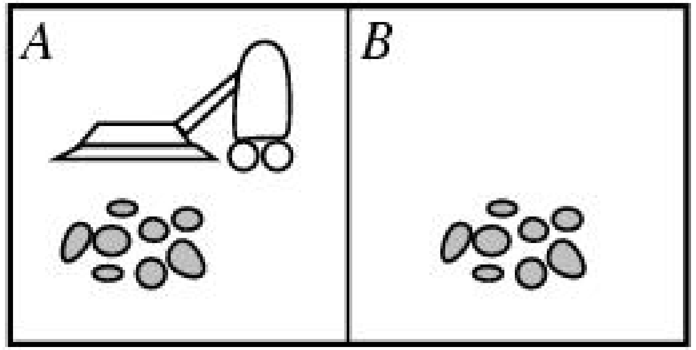
- Environment: square A and B
- Percepts: [location and content] e.g. *[A, Dirty]*
- Actions: left, right, suck dirt, and no-op

The vacuum-cleaner world



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
…	…

Implementing the Vacuum Cleaner Agent




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

- No doubt that this works for a small state space
- But how about a large state space?

A simple reflux agent

- The simplest type of agent is a *reflux* type
- A reflux agent takes an action <u>solely on the basis of the current state</u>
- A child who has mistakenly touched a hot stove will immediately withdraw its hand irrespective of what position it was before
- The vacuum cleaner will move to right to grid B irrespective of its previous position was either A or D, as long as either one was in a "clean" state

A	B
C	D

The concept of rationality

- A **rational agent** is one who behaves objectively – i.e., according to a well-defined rule or utility function
- Utility function must be objective
  - E.g. the amount of dirt cleaned within a certain time.
  - E.g. how clean the floor is.
  - The distance travelled in a path finder application
  - …
- Given a set of states S that the agent has stepped through, the next state is determined by a function *b(S,u),* where *u* is the utility function

Rationality

- Our behavior function b tells that rationality depends on four things:
  - Utility function
  - Environment knowledge,
  - The se of possible Actions,
  - Percept sequence of states already taken to date.
- DEF: *A rational agent chooses whichever action maximizes the expected value of the utility function given the percept sequence to date and prior environment knowledge.*

Rationality

- Rationality $\neq$ omniscience
  - An omniscient agent knows the actual outcome of its actions.
  - E.g. in a path finder application, we **do** know the outcome of visiting the next location if we apply a shortest path algorithm – we can guarantee that the decision is the optimal (best) one
- Rationality $\neq$ perfection
  - Rationality maximizes *expected* performance, while perfection maximizes *actual* performance.
- Unfortunately, in the real world some problems are too complex and specifying a perfect utility function is an impossibility, hence we have only approximately correct actions or behavior

Rationality

- Thus, our definition of rationality requires:
  - Information gathering/exploration
    - To maximize future rewards
  - Learn from percepts
    - Extending prior knowledge
  - Agent autonomy
    - Compensate for incorrect prior knowledge

Environments

- To design a rational agent, we must specify its task environment.
- PEAS description of the environment:
  - Performance
  - Environment
  - Actuators
  - Sensors

Environments

- **E.g. Fully automated taxi (see** [Waymo opens driverless robo-taxi service to the public in Phoenix | Reuters])
  - PEAS description of the environment:
    - Performance
      - Safety, destination, profit (what about legality and comfort level?)
    - Environment
      - Streets/freeways, other traffic, pedestrians, weather,, …
    - Actuators
      - Steering, accelerating, brake, horn, speaker/display,…
    - Sensors
      - Video, sonar, speedometer, engine sensors, keyboard, GPS, …

Environment types

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?				
Deterministic?				
Episodic?				
Static?				
Discrete?				
Single-agent?				

Solitaire is a single person card game – see Solitaire Card Game Rules - How to play Solitaire the card game

Bakgammon is a two person game: see Backgammon - Wikipedia

Environment types

Fully vs. partially observable: an environment is fully observable when the sensors can detect all aspects that are *relevant* to the choice of action.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?				
Deterministic?				
Episodic?				
Static?				
Discrete?				
Single-agent?				

Environment types

Fully vs. partially observable: an environment is full observable when the sensors can detect all aspects that are *relevant* to the choice of action.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?				
Episodic?				
Static?				
Discrete?				
Single-agent?				

Environment types

Deterministic vs. stochastic: if the next environment state is completely determined by the current state, then the environment is deterministic.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?				
Episodic?				
Static?				
Discrete?				
Single-agent?				

Environment types

Deterministic vs. stochastic: if the next environment state is completely determined by the current state the executed action then the environment is deterministic.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?	YES	NO	YES	NO
Episodic?				
Static?				
Discrete?				
Single-agent?				

Environment types

Episodic vs. sequential: In an episodic environment the agent's experience can be divided into atomic steps where the agents perceives and then performs a single action. The choice of action depends only on the episode itself

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?	YES	NO	YES	NO
Episodic?				
Static?				
Discrete?				
Single-agent?				

Environment types

Episodic vs. sequential: In an episodic environment the agent's experience can be divided into atomic steps where the agents perceives and then performs a single action. The choice of action depends only on the episode itself

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?	YES	NO	YES	NO
Episodic?	NO	NO	NO	NO
Static?				
Discrete?				
Single-agent?				

Environment types

Static vs. dynamic: If the environment can change while the agent is choosing an action, the environment is dynamic. Semi-dynamic if the agent's performance changes even when the environment remains the same.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?	YES	NO	YES	NO
Episodic?	NO	NO	NO	NO
Static?				
Discrete?				
Single-agent?				

Environment types

Static vs. dynamic: If the environment can change while the agent is choosing an action, the environment is dynamic. Semi-dynamic if the agent's performance changes even when the environment remains the same.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?	YES	NO	YES	NO
Episodic?	NO	NO	NO	NO
Static?	YES	YES	SEMI	NO
Discrete?				
Single-agent?				

Environment types

Discrete vs. continuous: This distinction can be applied to the state of the environment, the way time is handled and to the percepts/actions of the agent.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	**FULL**	**FULL**	**PARTIAL**	**PARTIAL**
Deterministic?	**YES**	**NO**	**YES**	**NO**
Episodic?	**NO**	**NO**	**NO**	**NO**
Static?	**YES**	**YES**	**SEMI**	**NO**
Discrete?				
Single-agent?				

Environment types

Discrete vs. continuous: This distinction can be applied to the state of the environment, the way time is handled and to the percepts/actions of the agent.

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable??	**FULL**	**FULL**	**PARTIAL**	**PARTIAL**
Deterministic??	**YES**	**NO**	**YES**	**NO**
Episodic??	**NO**	**NO**	**NO**	**NO**
Static??	**YES**	**YES**	**SEMI**	**NO**
Discrete??	**YES**	**YES**	**YES**	**NO**
Single-agent??				

Environment types

Single vs. multi-agent: Does the environment contain other agents who are also maximizing some performance measure that depends on the current agent's actions?

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	**FULL**	**FULL**	**PARTIAL**	**PARTIAL**
Deterministic?	**YES**	**NO**	**YES**	**NO**
Episodic?	**NO**	**NO**	**NO**	**NO**
Static?	**YES**	**YES**	**SEMI**	**NO**
Discrete?	**YES**	**YES**	**YES**	**NO**
Single-agent?				

Environment types

Single vs. multi-agent: Does the environment contain other agents who are also maximizing some performance measure that depends on the current agent's actions?

	Solitaire	Backgammom	Intenet shopping	Taxi
Observable?	FULL	FULL	PARTIAL	PARTIAL
Deterministic?	YES	NO	YES	NO
Episodic?	NO	NO	NO	NO
Static?	YES	YES	SEMI	NO
Discrete?	YES	YES	YES	NO
Single-agent?	YES	NO	NO	NO

Environment types

- The simplest environment is:
  - Fully observable, deterministic, episodic, static, discrete and single-agent.
- Most real situations are:
  - Partially observable, stochastic, sequential, dynamic, continuous and multi-agent.

Agent types

- What do the agents consist of?
  - Agent = architecture + program

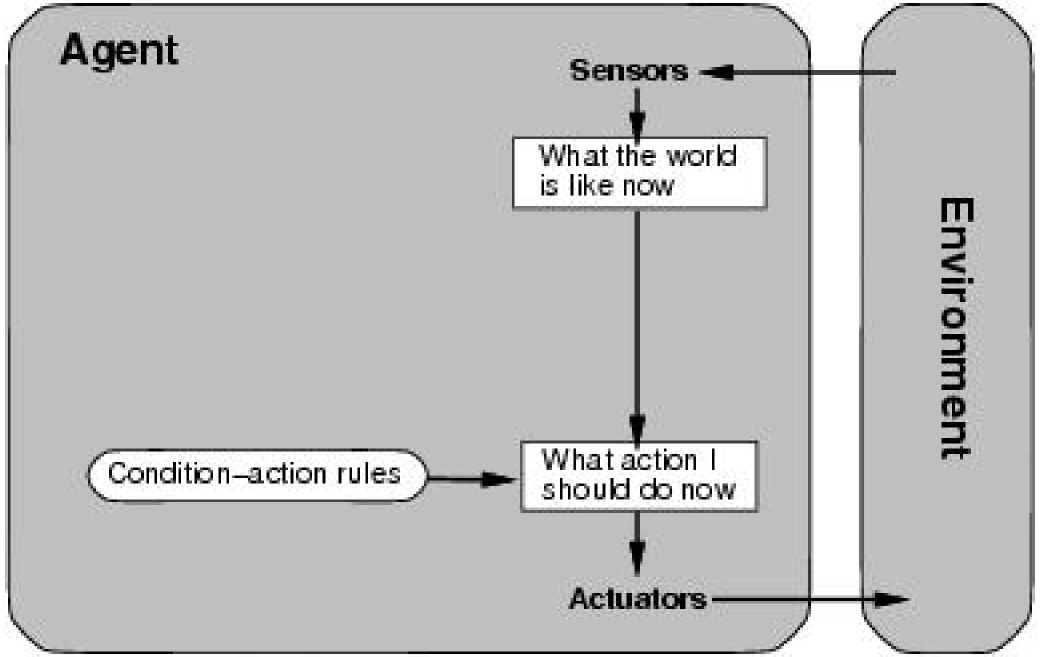
- All agents have the same skeleton:
  - Input = current percepts
  - Output = action
  - Program= manipulates input to produce output

- Of course, different agents have different utility functions which serve the individual needs of the application they are servicing

Agent types

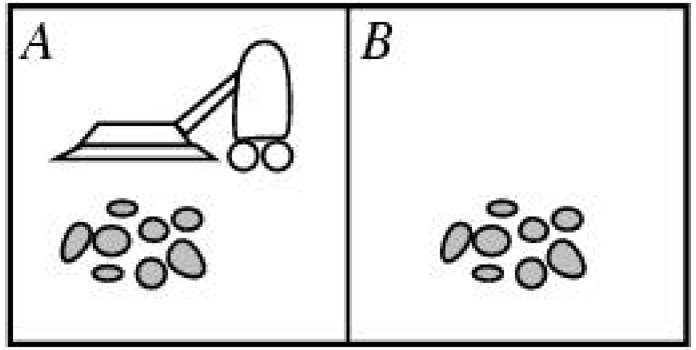
- Four basic kind of agent agents will be discussed:
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based agents
- All these can be turned into learning agents.

The simple reflex agent type



- Select action on the basis of *only the current* percept.
  - E.g. the vacuum-agent
- Large reduction in possible state space
- Implemented through *condition-action rules*
  - `If dirty then suck`

The vacuum-cleaner world




```
function REFLEX-VACUUM-AGENT ([location, status])
  return an action

  if status == Dirty then return Suck

  else if location == A then return Right

  else if location == B then return Left
```

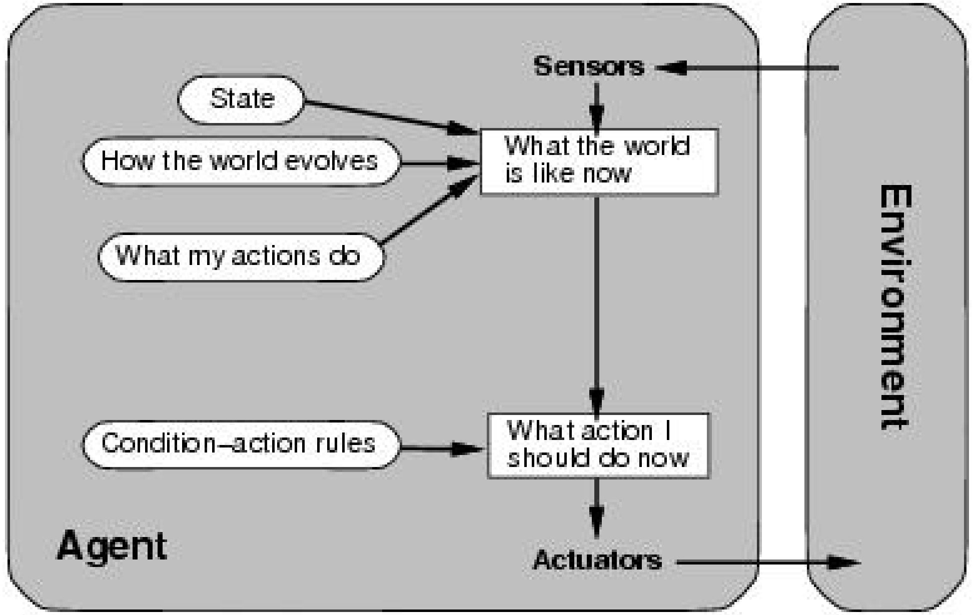
- We achieve Reduction in state space from 4^T to 4 entries but pay a big price for this

The simple reflex agent

```
function SIMPLE-REFLEX-AGENT(percept) returns an action

  static: rules, a set of condition-action rules

  state ← INTERPRET-INPUT(percept)
  rule ← RULE-MATCH(state, rule)
  action ← RULE-ACTION[rule]
  return action
```

- Will only work if the environment is *fully observable* otherwise infinite loops may occur.

Agent types: reflex and state



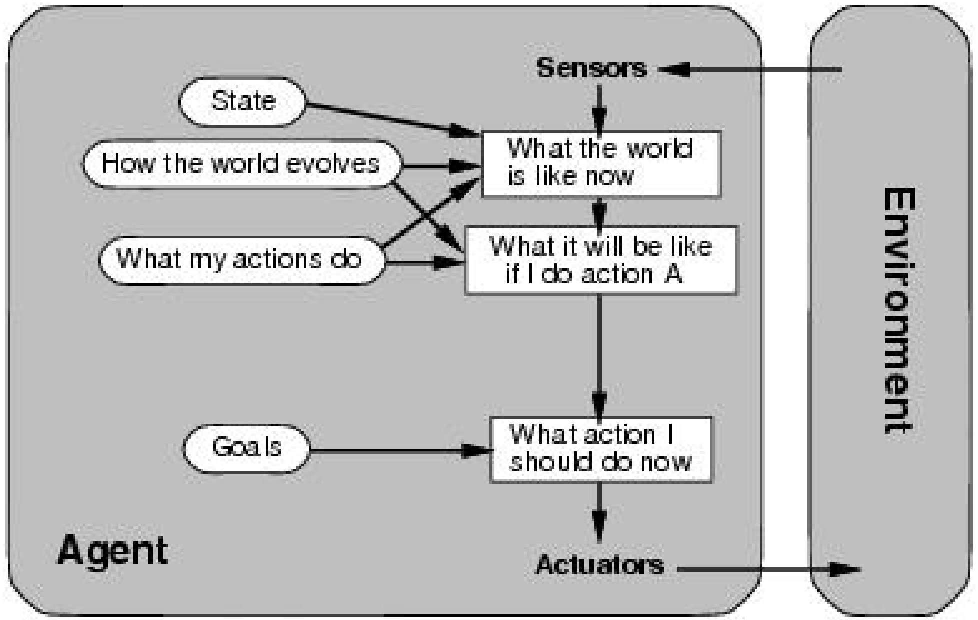
- To tackle *partially observable* environments.
  - Maintain internal state
- Over time update state using world knowledge
  - How does the world change.
  - How do actions affect world.

  $\Rightarrow$ *Model of World*

Agent types: reflex and state

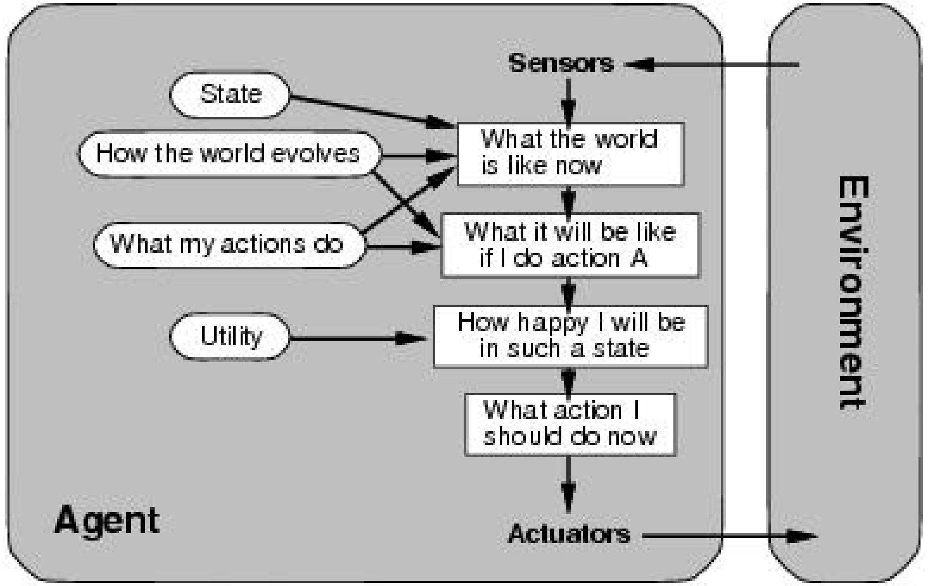
```
function REFLEX-AGENT-WITH-STATE(percept) returns an action

  static: rules, a set of condition-action rules
          state, a description of the current world state
          action, the most recent action.

  state ← UPDATE-STATE(state, action, percept)
  rule ← RULE-MATCH(state, rule)
  action ← RULE-ACTION[rule]
  return action
```

Agent types: goal-based



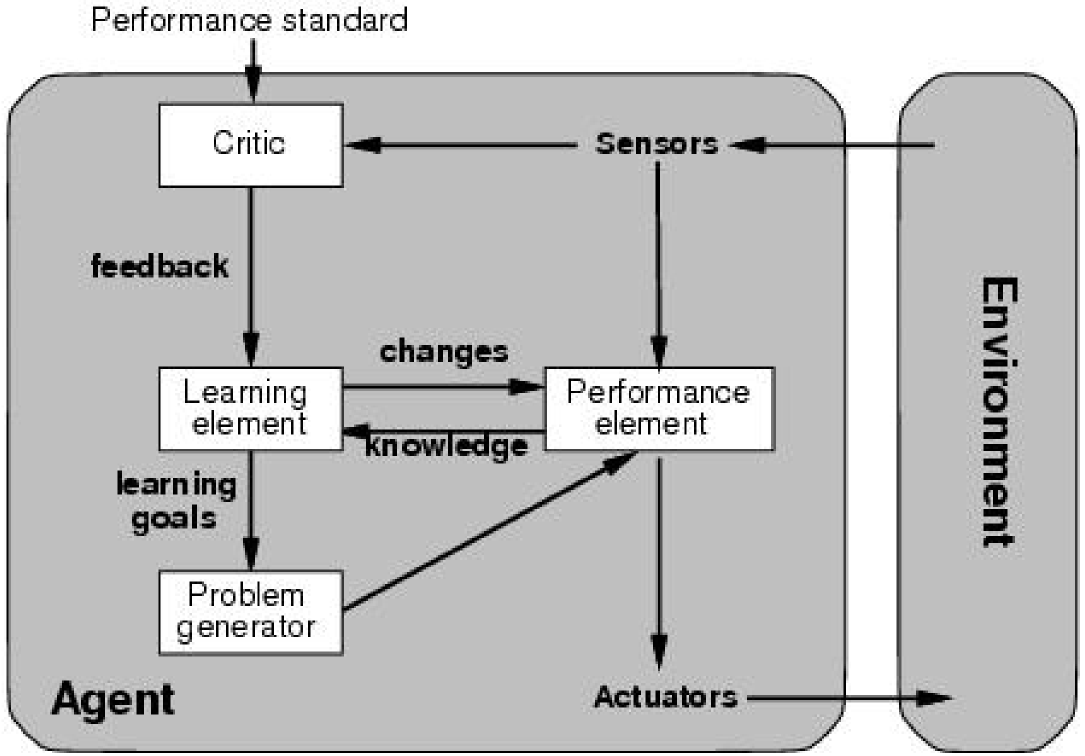
- The agent needs a goal to know which situations are *desirable*.
  - Things become difficult when long sequences of actions are required to find the goal.
- Typically investigated in **search** and **planning** research.
- Major difference: future is taken into account
- Is more flexible since knowledge is represented explicitly and can be manipulated.

Agent types: utility-based



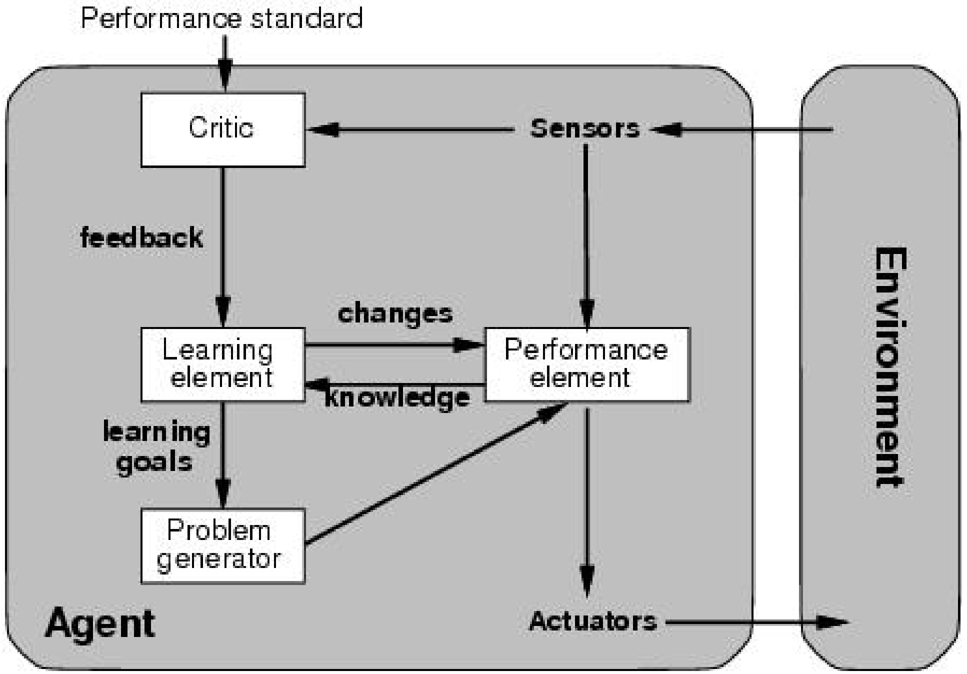
- Certain goals can be reached in different ways.
  - Some are better, have a higher utility.
- Utility function maps a (sequence of) state(s) onto a real number.
- Improves on goals:
  - Selecting between conflicting goals
  - Select appropriately between several goals based on likelihood of success.

Agent types: Learning



- All previous agent-programs describe methods for selecting *actions*.
  - Yet it does not explain the origin of these programs.
  - Learning mechanisms can be used to perform this task.
  - Teach them instead of instructing them.
  - Advantage is the robustness of the program toward initially unknown environments.

Agent types: Learning



- *Learning element*: introduce improvements in performance element.
  - Critic provides feedback on agent's performance based on fixed performance standard.
- *Performance element*: selecting actions based on percepts.
  - Corresponds to the previous agent programs
- *Problem generator*: suggests actions that will lead to new and informative experiences.
  - Exploration vs. exploitation

Summary

- Performance
- Environment
- Actuators
- Sensors

- Observable
- Static/(semi)dynamic
- Deterministic/Stochastic
- Episodic/sequential
- Discrete/continuous
- Single/Multiple agent

Reflex model

Simple reflex

Table-driven reflex

Goal-driven model

Utility-driven model

Learning model