Survey

I **Artificial Intelligence**

1. Introduction

2. Intelligent Agents

II Problem Solving

III Knowledge, Reasoning, Planning

IV Uncertain Knowledge and Reasoning

V Machine Learning

VI Communicating, Perceiving, and Acting

VII Conclusions

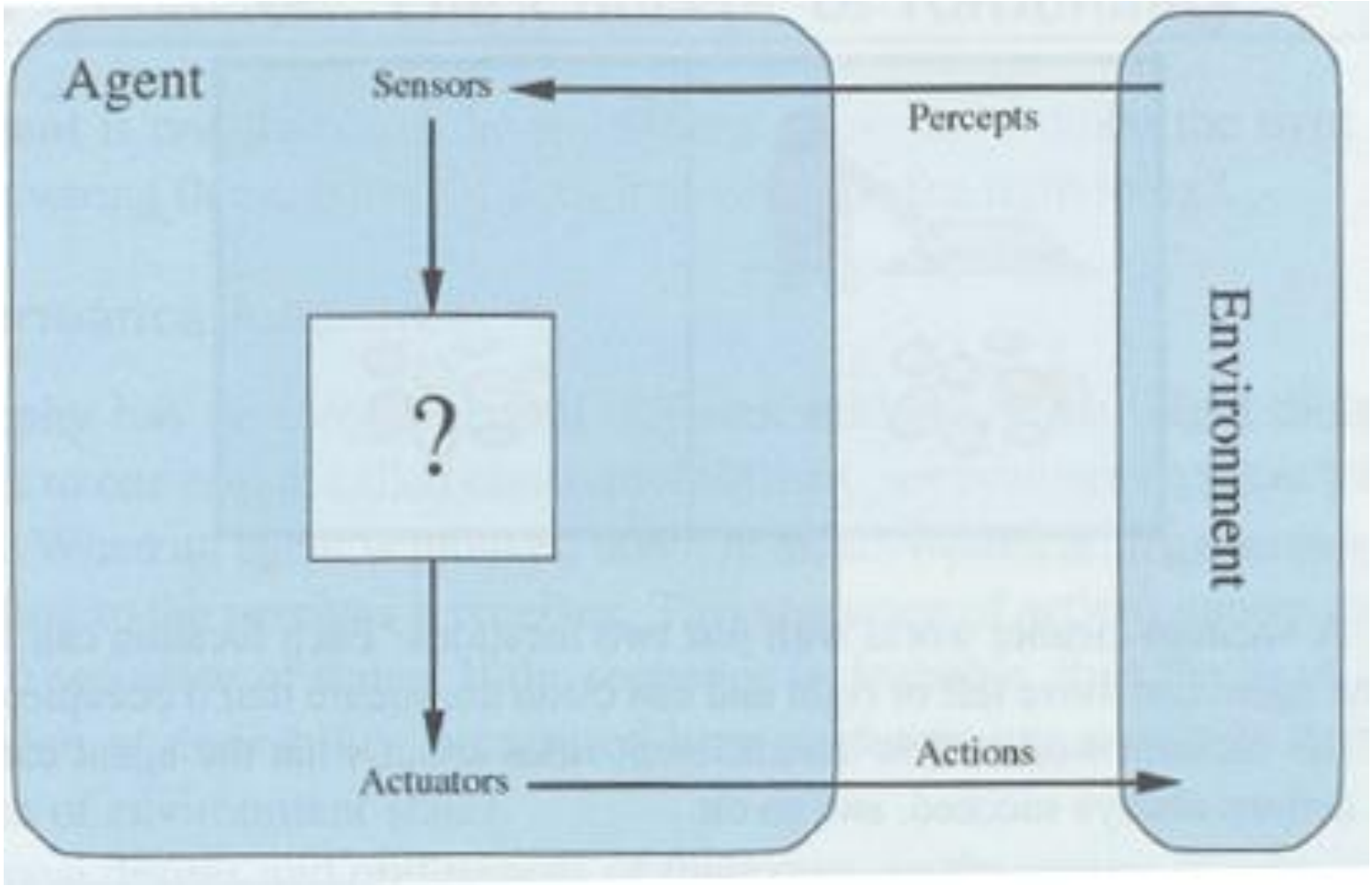
Intelligent Agents

- Agents and Environments
- Performance Measure
- Properties of Environments
- The Structure of Agents

Agents and Environments

Agent function:

Mapping of percept history on actions: **f: P* → A**



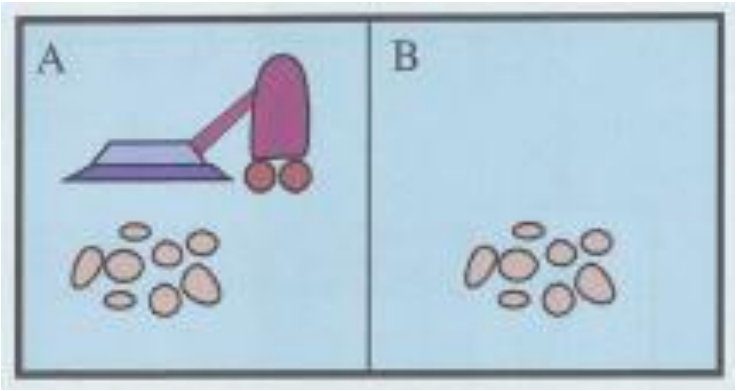
Sensors and Actuators for Agents

- **Humans:** *Sensors:* sensory organs
- *Actuators:* body parts with muscles

- **Robots:** *Sensors*: camera, infrarot, sonar etc.
- *Actuators:* wheels, joints, etc. with motors

- …

- **vacuum cleaner:** *Sensors = percepts:* location & dirt
- *Actuators = Actions:* Left, Right, Suck, NoOp



Simple agent function

Rule:

if the current square is dirty,

then suck

else move to other field

Table on percept sequences**:**

(table on last percept smaller)

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
⋮	⋮
[A, Clean], [A, Clean], [A, Clean]	Right
[A, Clean], [A, Clean], [A, Dirty]	Suck
⋮	⋮

Performance Measure

- **Performance measure should be carefully defined:**
  - Vacuum cleaner: amount of dirt vs. clean floor?
    - Warning: agent might bring some dirt, distribute and suck it
  - Clean floor: cleaning either often mediocre or rarely very good?
  - Amount of time and energy relevant?

Concept of Rationality

- **Definition of a 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.

- **Analysis:**
  - does not require **omniscience**
  - might require **information gathering** (e.g. when crossing a road, you should look left and right, but not above) or **exploration** of unknown environments
  - might require **learning** from percept sequences

"PEAS"-Task environment of agents

- Performance Criterion
- Environment
- Actuators
- Sensors

- Example for PEAS of taxi driver:

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits, minimize impact on other road users	Roads, other traffic, police, pedestrians, customers, weather	Steering, accelerator, brake, signal, horn, display, speech	Cameras, radar, speedometer, GPS, engine sensors, accelerometer, microphones, touchscreen

Examples for Agent types and PEAS-descriptions

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments	Touchscreen/voice entry of symptoms and findings
Satellite image analysis system	Correct categorization of objects, terrain	Orbiting satellite, downlink, weather	Display of scene categorization	High-resolution digital camera
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, tactile and joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, raw materials, operators	Valves, pumps, heaters, stirrers, displays	Temperature, pressure, flow, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, feedback, speech	Keyboard entry, voice

Properties of task environments

	Vaccum cleaner:
Fully observable vs. partially observable	Fully observable
Deterministic vs. stochastic	deterministic
Episodic vs. sequential	sequential
Static vs. dynamic	static
Discrete vs. continuous	discrete
Single agent vs. multiagent	single agent
Known vs. unknown	known

Examples of task environments

	Solitaire	Backgammon	Internet-Shopping	Taxi
observable?	?	?	?	?
Deterministic?	?	?	?	?
Episodic?	?	?	?	?
Static?	?	?	?	?
Discrete?	?	?	?	?
Sinlge-Agent?	?	?	?	?
Known?	?	?	?	?

Examples of task environments

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
English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Environment Simulator

- For agent programming an environment simulator including an environment generator is necessary.
- The performance of an agent might depend strongly on the environment.
- Example: Civilization (game)

Structure of Agents

- **Agent** = Architecture + Program
  - **Architecture:** Hardware (sensors, computer-hardware, actuators, communication channels)
  - **Program:** table, rules, searching, … ???

- **Autonomous Agent**
  - weak Autonomy
  - strong Autonomy
  - combination

Agent Function as Table?

- It is theoretically possible to describe an agent with a table, that maps possible percept sequences to actions.
- Problem: very large or infinite tables
- Example: Part of square root table and alternate description as function (right)

Percept x	Action z
1.0	1.000000000000000
1.1	1.048808848170152
1.2	1.095445115010332
1.3	1.140175425099138
1.4	1.183215956619923
1.5	1.224744871391589
1.6	1.264911064067352
1.7	1.303840481040530
1.8	1.341640786499874
1.9	1.378404875209022
⋮	⋮

function SQRT(x)
  $z \leftarrow 1.0$ /* *initial guess* */
  repeat until $|z^2 - x| < 10^{-15}$
    $z \leftarrow z - (z^2 - x)/(2z)$
  end
  return z

Structure of table driven agent

function TABLE-DRIVEN-AGENT(*percept*) **returns** *action*

static: *percepts*, a sequence, initially empty

table, a table, indexed by percept sequences, initially fully specified

append *percept* to the end of *percepts*

action ← LOOKUP(*percepts*, *table*)

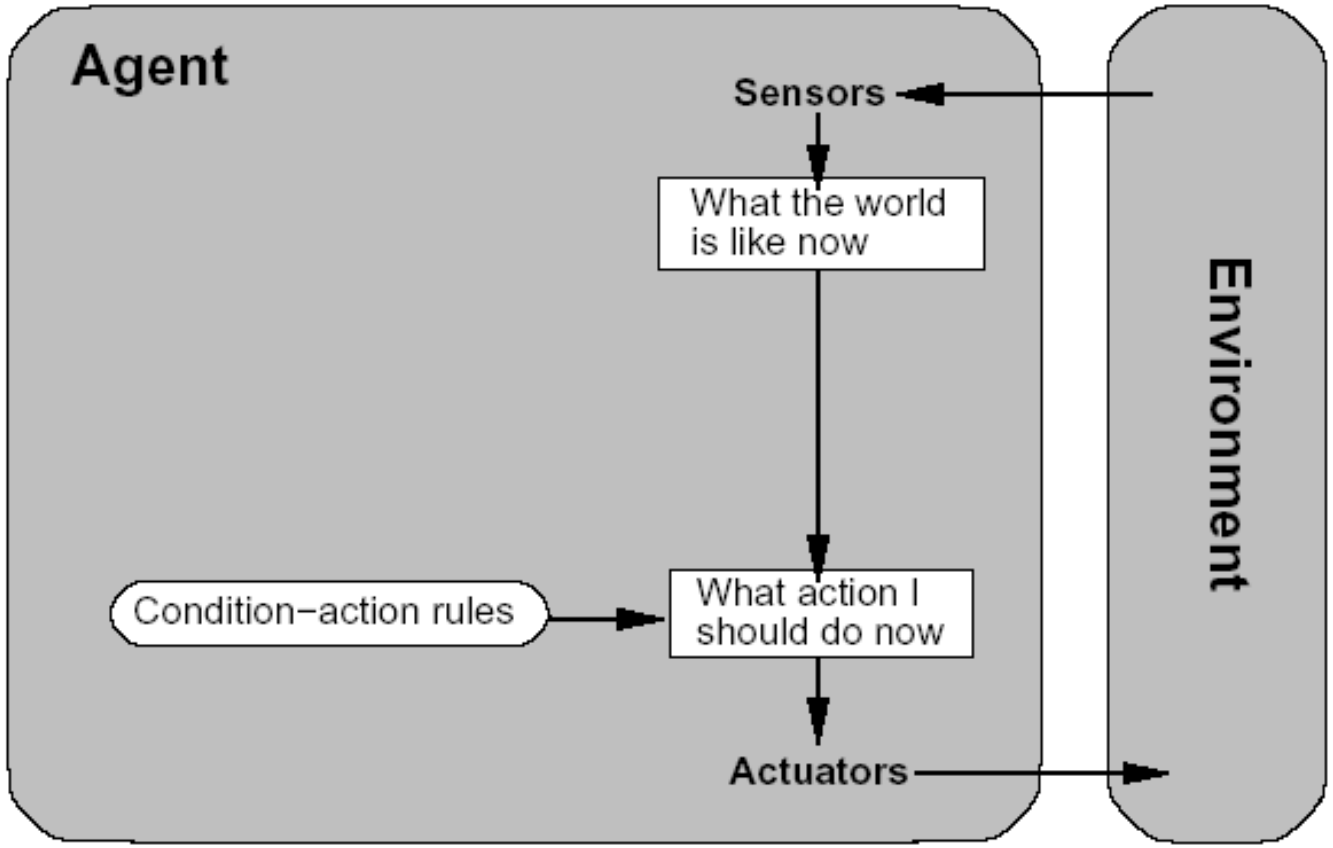
return *action*

Further types of Agents

- (Table-based agents)
- **Simple reflex agents:** Rules without memory
- **Model-based reflex agents:** Rules with memory
- **Goal-based agents:** Search or planning to achieve goal
- **Utility-based agents:** Balancing probabilities and utilities

- **Degree of Learning**

Simple Reflex Agent



function Simple Reflex Agent (*percept*)
returns an action
persistent: *rules*, a set of condition-action rules

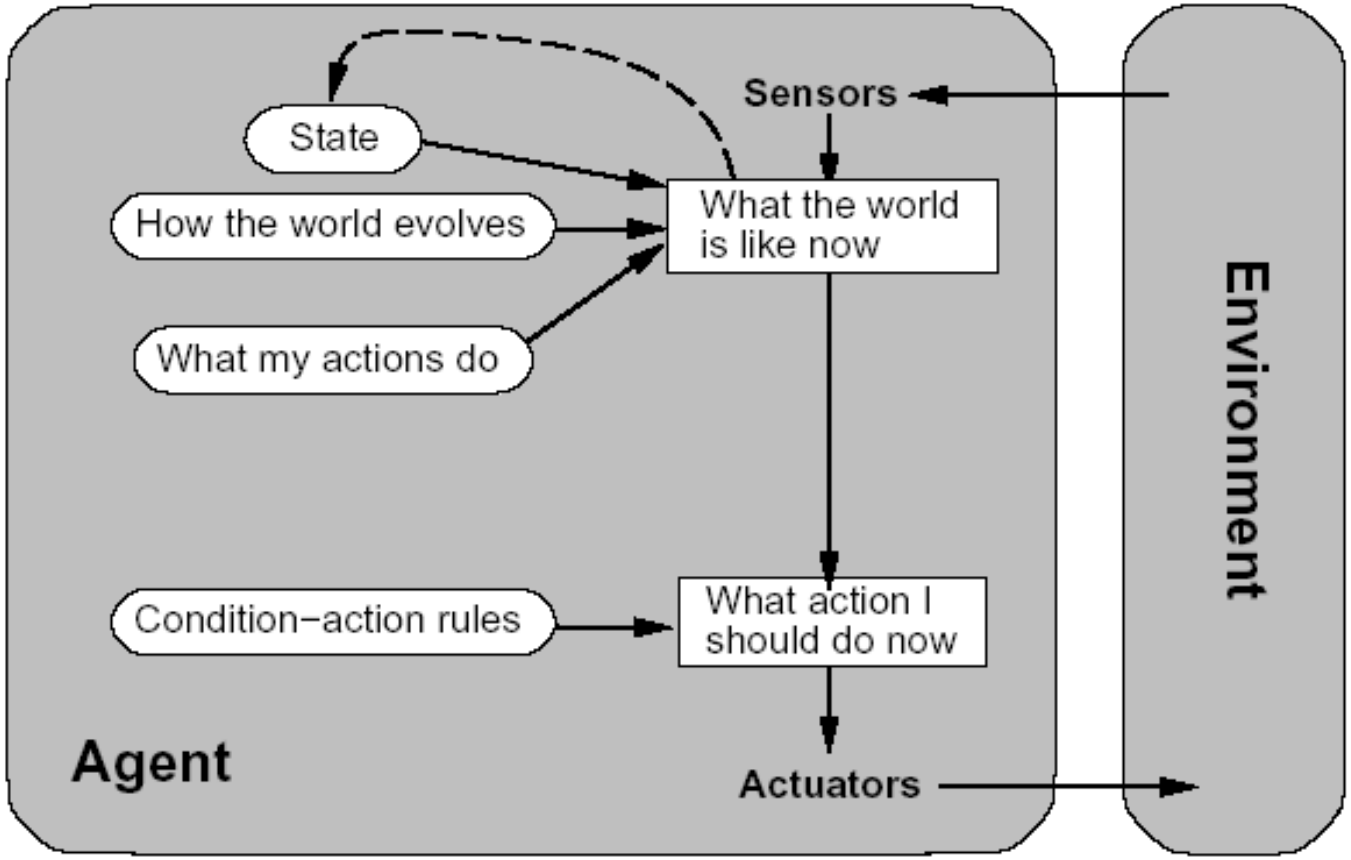
state ← Interpret-Input (*percept*)
rule ← Rule-Match (*state, rules*)
action ← *rule*. Action
return *action*

Model-based Reflex Agent

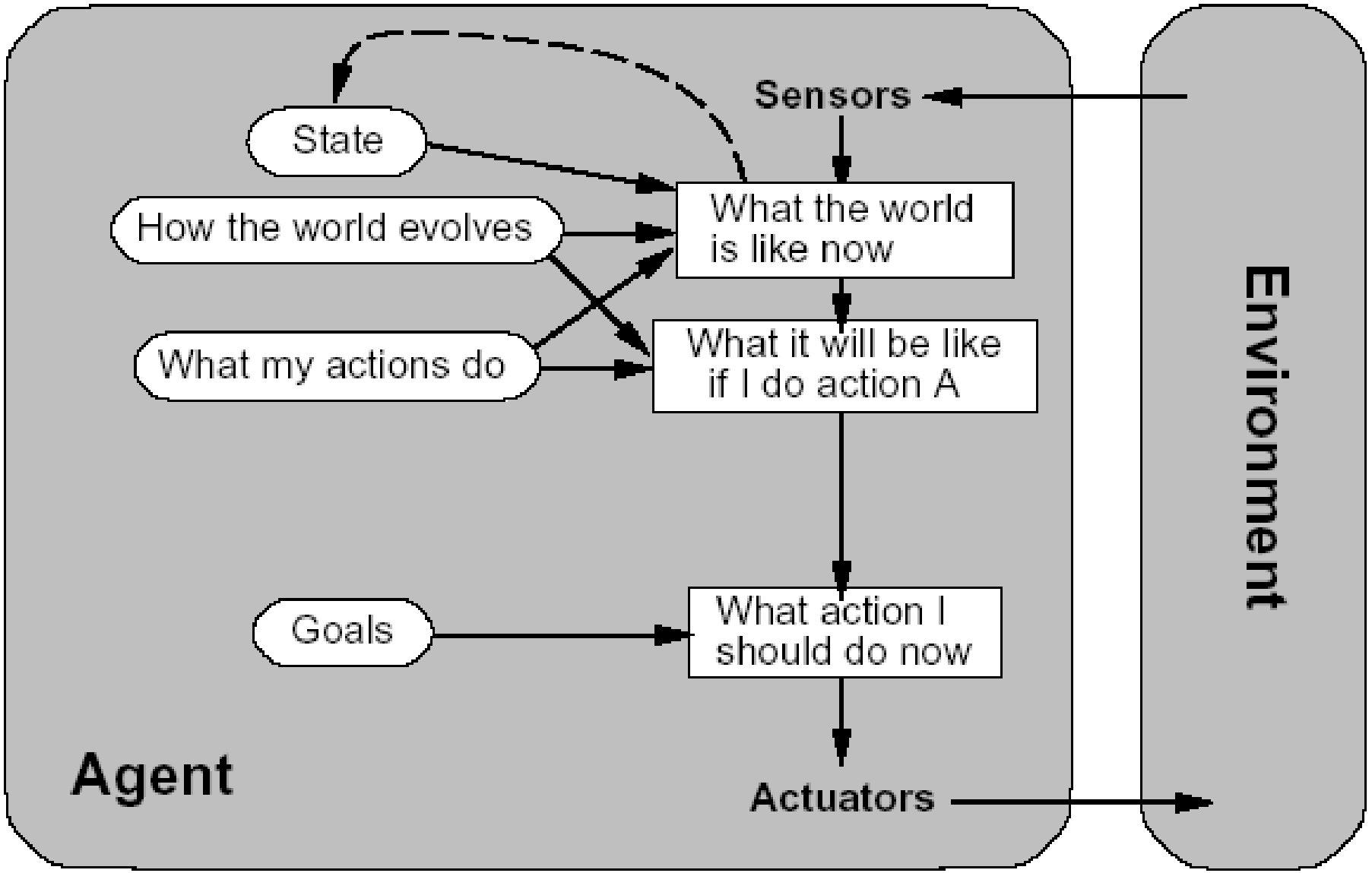
function Model-Based-Reflex Agent (*percept*) **returns** an action

persistent: *state*: the agent‘s current conception of the world state
transition-model: a decription of how the next state depends on current state & action
sensor-model: a description of how the current world state ist reflected in the agent‘s perception
rules: a set of condition-action rules
action: the most recent action, initially none

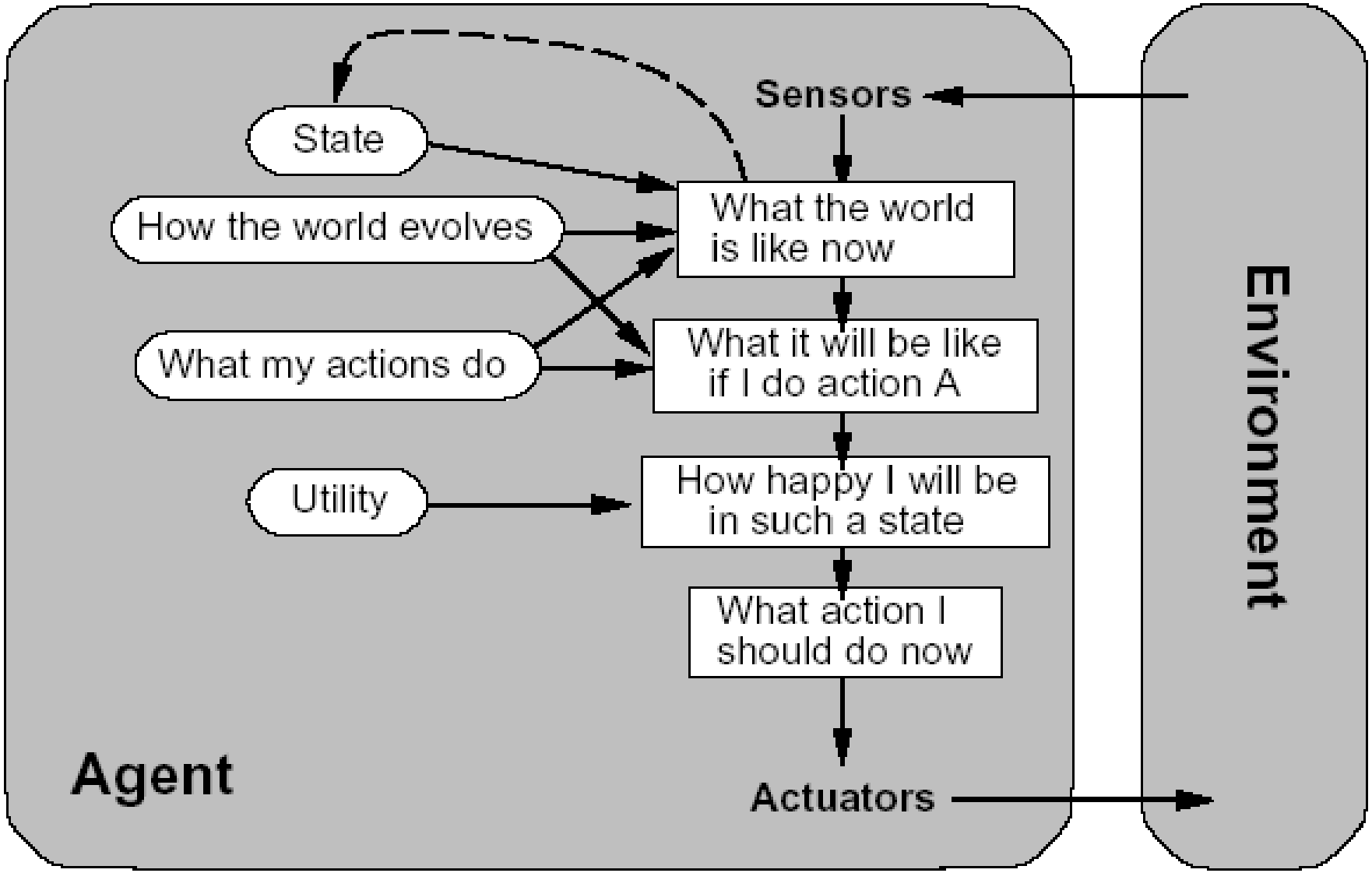
state ← Update-State (state, action, percept, transition-model, sensor-model)
rule ← Rule-Match (state, rules)
action ← rule. Action
return *action*



Goal-based Agent



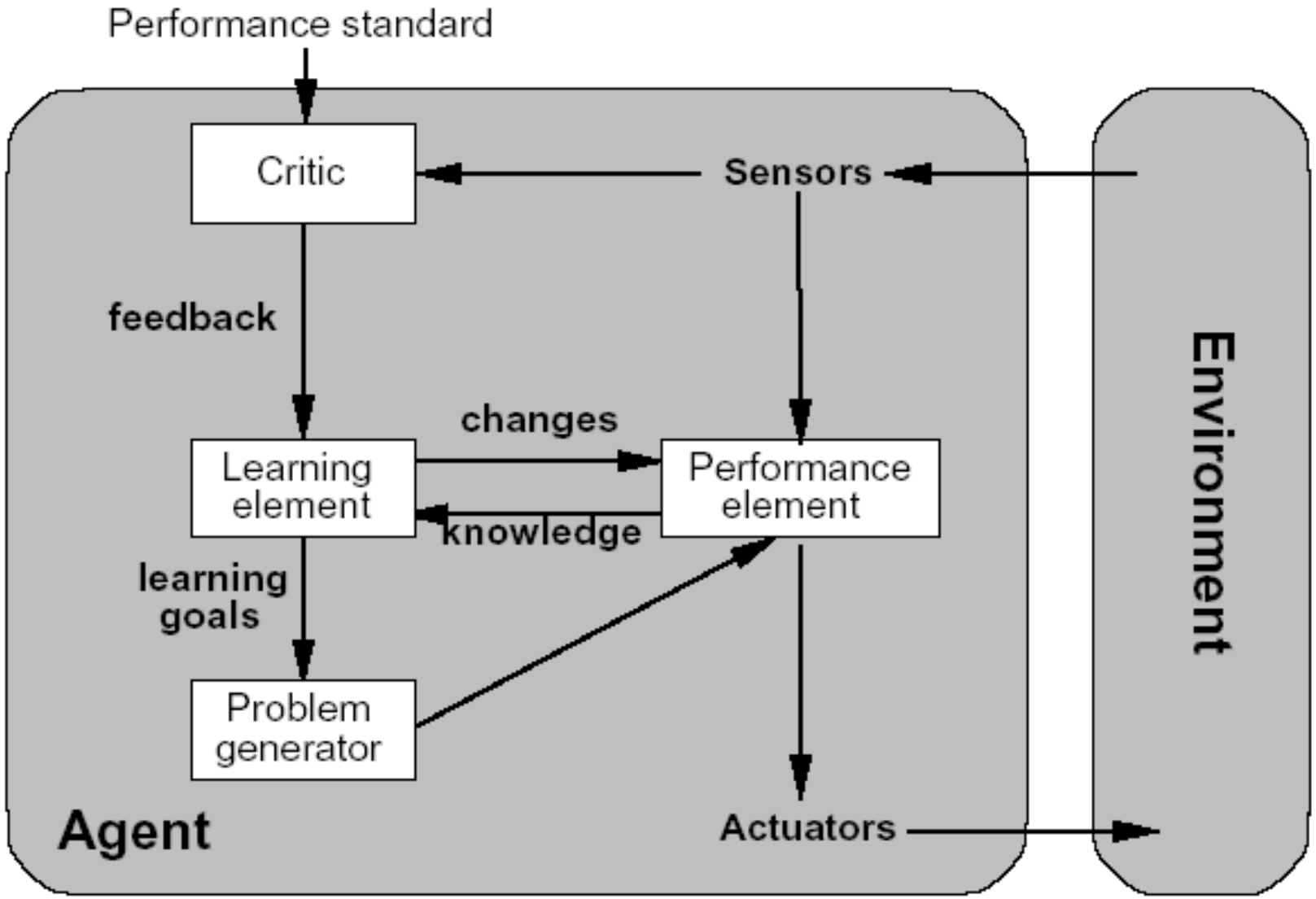
Utility-based Agent



Goal- vs. Utility-based Agents

- With just one discrete goal not distinguishable
- With multiple goals utility-based agents offer more expressiveness
  - comparative assessment of goal conflicts (e.g. time vs. safety of a travelling route)
  - comparative assessment between reachability vs. importance among different, potential unreachable goals

Learning Agent



Components:

- **Critic element:** Compare externally given performance standard with percepts and gives feedback to learning element
- **Learning element:** Changes (knowledge of) performance element
- **Problem generator:** Suggests actions leading to new situations to create learning opportunities

Types of State Representations

- **Atomic** (Black Box)
- **Factored** (State as attribute-value vector)
- **Structured** (State as objects with relations)