Intelligent Agents

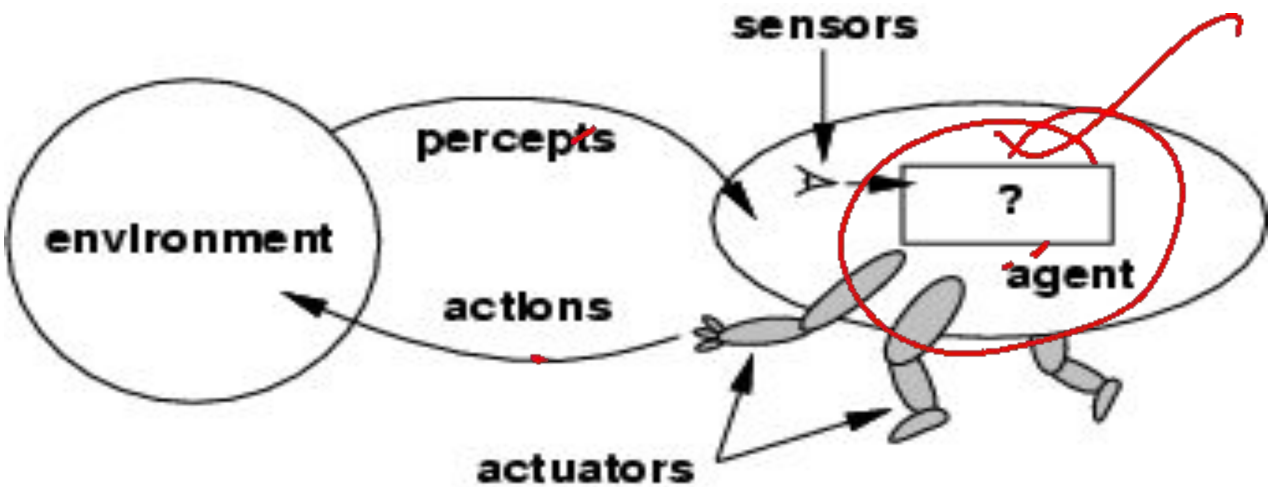
Outline

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

Agents

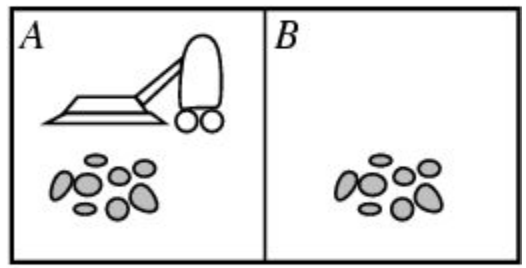
- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
- Human agent:
  - eyes, ears, and other organs for sensors;
  - hands, legs, mouth, and other body parts for actuators
- Robotic agent:
  - cameras and infrared range finders for sensors;
  - various motors for actuators

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

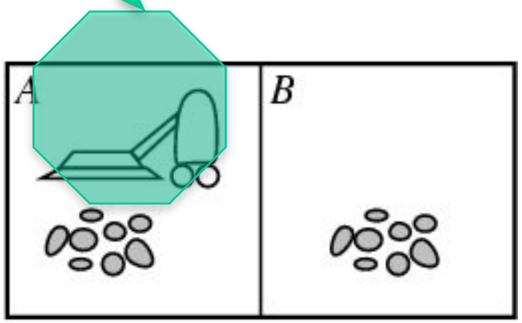
Vacuum-cleaner world



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

Agent / Robot

E.g., vacuum-cleaner world



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

Actions: *Left*, *Right*, *Suck*, *NoOp*

iRobot Roomba® 400
Vacuum Cleaning Robot

iRobot Corporation

Founder Rodney Brooks (MIT)

- Powerful suction and rotating brushes
- Automatically navigates for best cleaning coverage
- Cleans under and around furniture, into corners and along wall edges
- Self-adjusts from carpets to hard floors and back again
- Automatically avoids stairs, drop-offs and off-limit areas
- Simple to use— just press the Clean button and Roomba does the rest

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 eg crossing the road.)
- 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)

Task Environments

- Task Environments is the problems to which rational agents are the solutions.
- For a Vacuum-cleaner agent we need to specify the performance measure, the environment, and the agent’s sensors and actuators.
- This is called as the PEAS description.

PEAS

- 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:

  – Performance measure
  – Environment
  – Actuators
  – Sensors

Characterizing a Task Environment

Must first specify the setting for intelligent agent design.

PEAS: Performance measure, Environment, Actuators, Sensors

Example: the task of designing a self-driving car

- **Performance measure** **Safe, fast, legal, comfortable trip**
- **Environment** **Roads, other traffic, pedestrians**
- **Actuators** **Steering wheel, accelerator, brake, signal, horn**
- **Sensors** **Cameras, LIDAR (light/radar), speedometer, GPS, odometer engine sensors, keyboard**

PEAS

- Must first specify the setting for intelligent 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

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)

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

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

Task Environment

- Task Environments is the problems to which rational agents are the solutions.

Environment types

- 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 types

- 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.

Task Environments

1) Fully observable / Partially observable

– If an agent's sensors give it access to the complete state of the environment needed to choose an action, the environment is fully observable.

(e.g. chess – what about Kriegspiel?)

2) Deterministic / Stochastic

- An environment is deterministic if the next state of the environment is completely determined by the current state of the environment and the action of the agent;
- In a stochastic environment, there are multiple, unpredictable outcomes. (If the environment is deterministic except for the actions of other agents, then the environment is strategic).

In a fully observable, deterministic environment, the agent need not deal with uncertainty.

Note: Uncertainty can also arise because of computational limitations. E.g., we may be playing an omniscient ("all knowing") opponent but we may not be able to compute his/her moves.

3) Episodic / Sequential

- In an episodic environment, the agent's experience is divided into atomic episodes. Each episode consists of the agent perceiving and then performing a single action.

- Subsequent episodes do not depend on what actions occurred in previous episodes. Choice of action in each episode depends only on the episode itself. (E.g., classifying images.)

- In a sequential environment, the agent engages in a series of connected episodes. Current decision can affect future decisions. (E.g., chess and driving)

4) Static / Dynamic

- A static environment does not change while the agent is thinking.

- The passage of time as an agent deliberates is irrelevant.

- The environment is semidynamic if the environment itself does not change with the passage of time but the agent's performance score does.

5) Discrete / Continuous

- **If the number of distinct percepts and actions is limited, the environment is discrete, otherwise it is continuous.**

6) Single agent / Multi-agent

- **If the environment contains other intelligent agents, the agent needs to be concerned about strategic, game-theoretic aspects of the environment (for either cooperative *or* competitive agents).**
- **Most engineering environments don't have multi-agent properties, whereas most social and economic systems get their complexity from the interactions of (more or less) rational agents.**

Environment types

	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

Agent functions and programs

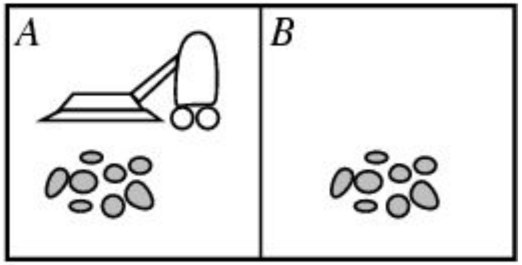
- An agent is completely specified by the agent function mapping percept sequences to actions
- One agent function (or a small equivalence class) is rational
- Aim: find a way to implement the rational agent function concisely

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

Toy example: Vacuum world.



Percepts: robot senses it's location and "cleanliness."

So, location and contents, e.g., [A, Dirty], [B, Clean].

With 2 locations, we get **4 different possible sensor inputs.**

Actions: *Left*, *Right*, *Suck*, *NoOp*

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*
⋮	⋮

Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

Table lookup

Action sequence of length K, gives 4^K different possible sequences. At least many entries are needed in the table. So, even in this very toy world, with K = 20, you need a table with over 4^20 > 10^12 entries.

In more real-world scenarios, one would have many more different percepts (eg many more locations), e.g., >=100. There will therefore be 100^K different possible sequences of length K. For K = 20, this would require a table with over 100^20 = 10^40 entries. Infeasible to even store.

So, table lookup formulation is mainly of theoretical interest. For practical agent systems, we need to find much more compact representations. For example, logic-based representations, Bayesian net representations, or neural net style representations, or use a different agent architecture, e.g., "ignore the past" --- Reflex agents.

Agent program for a vacuum-cleaner agent

Function TABLE-DRIVEN-AGENT(percept) returns an action

persistent: percepts,a sequence, initially empty

table,a table of actions, indexed by percept sequence, initially fully specified.

append the percept at the end of percepts

action<-lookup(percepts,table)

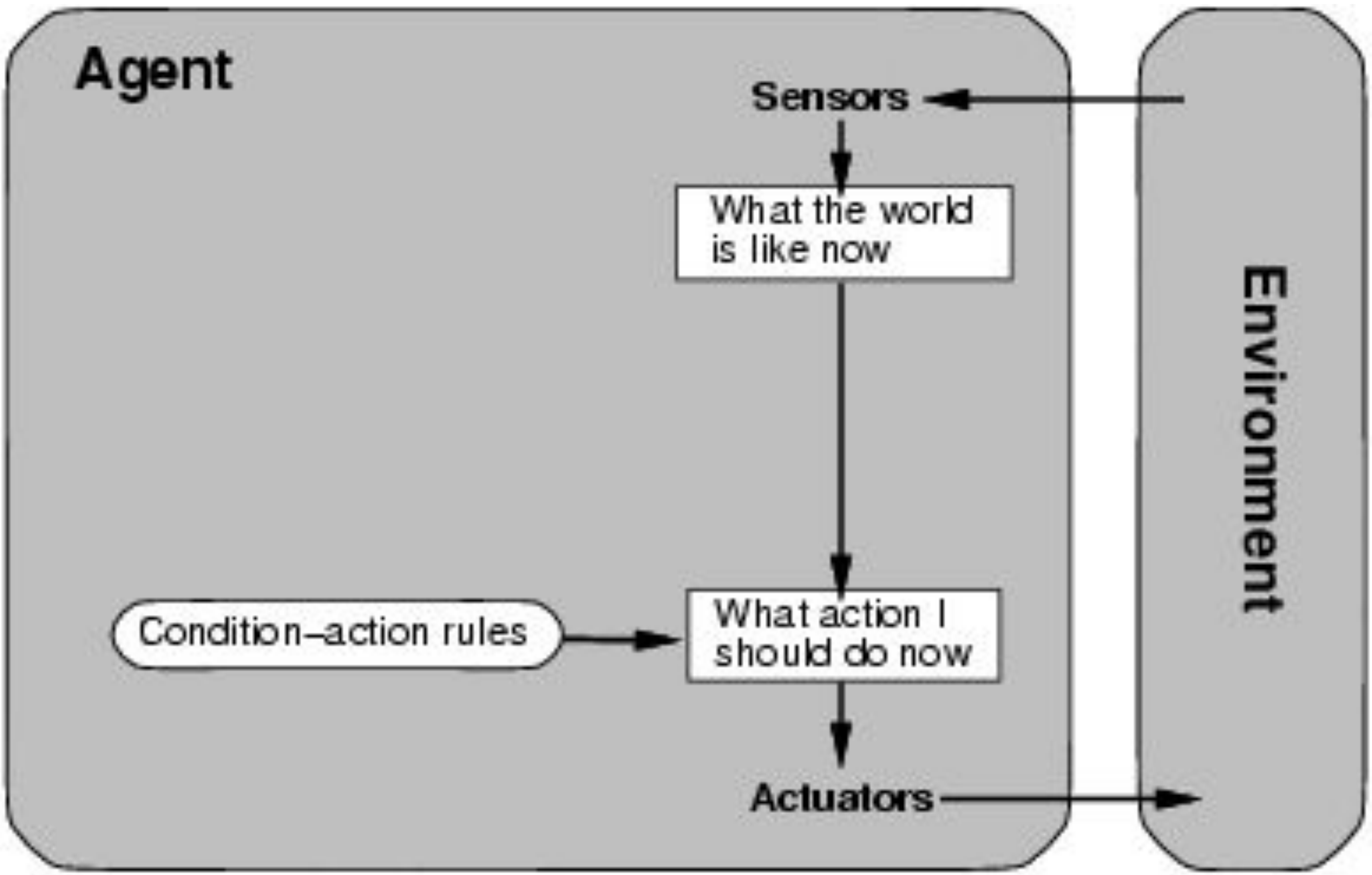
return action

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

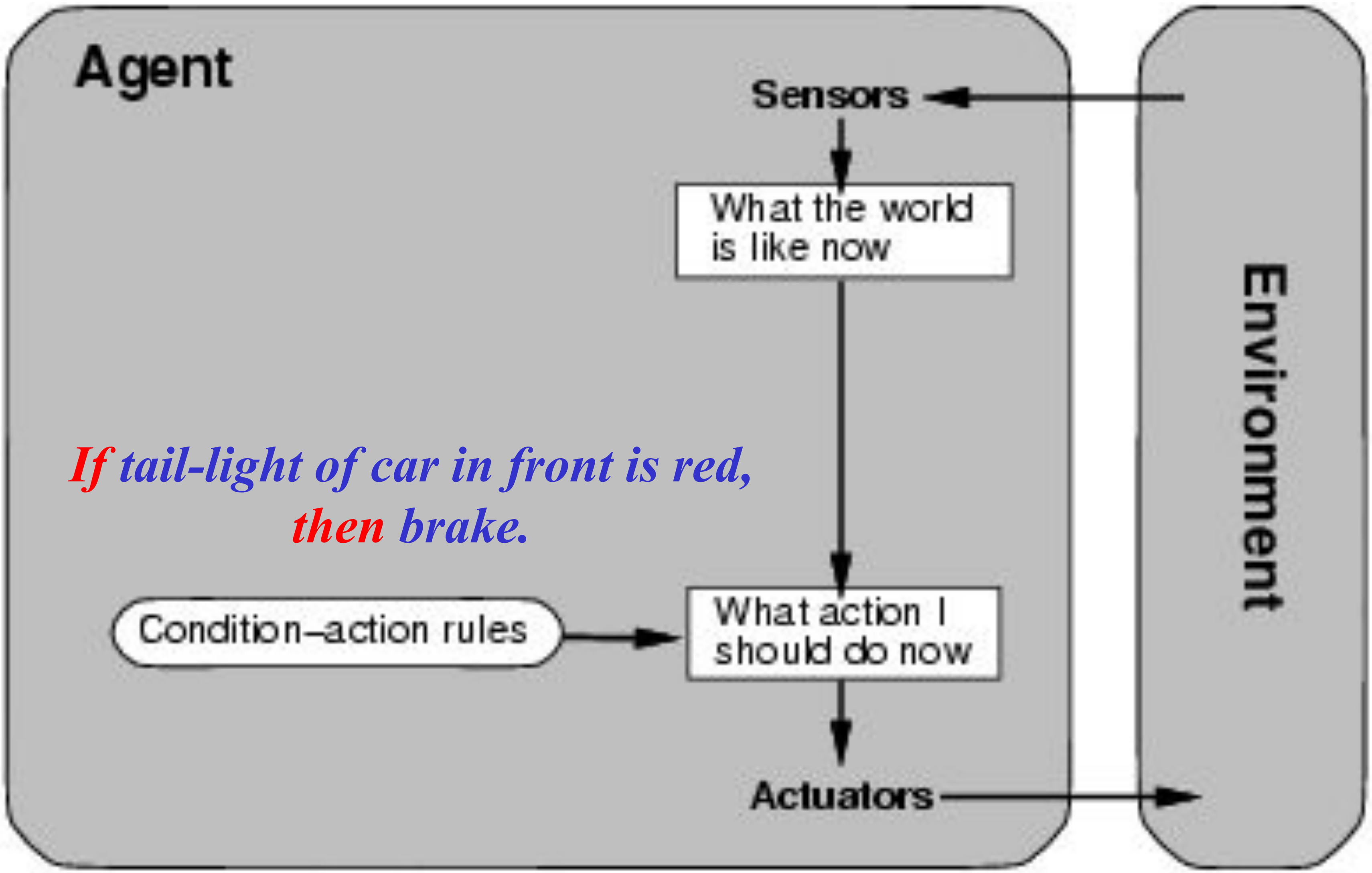
Agents **do not have memory** of past world states or percepts.

So, actions depend solely on **current percept**.

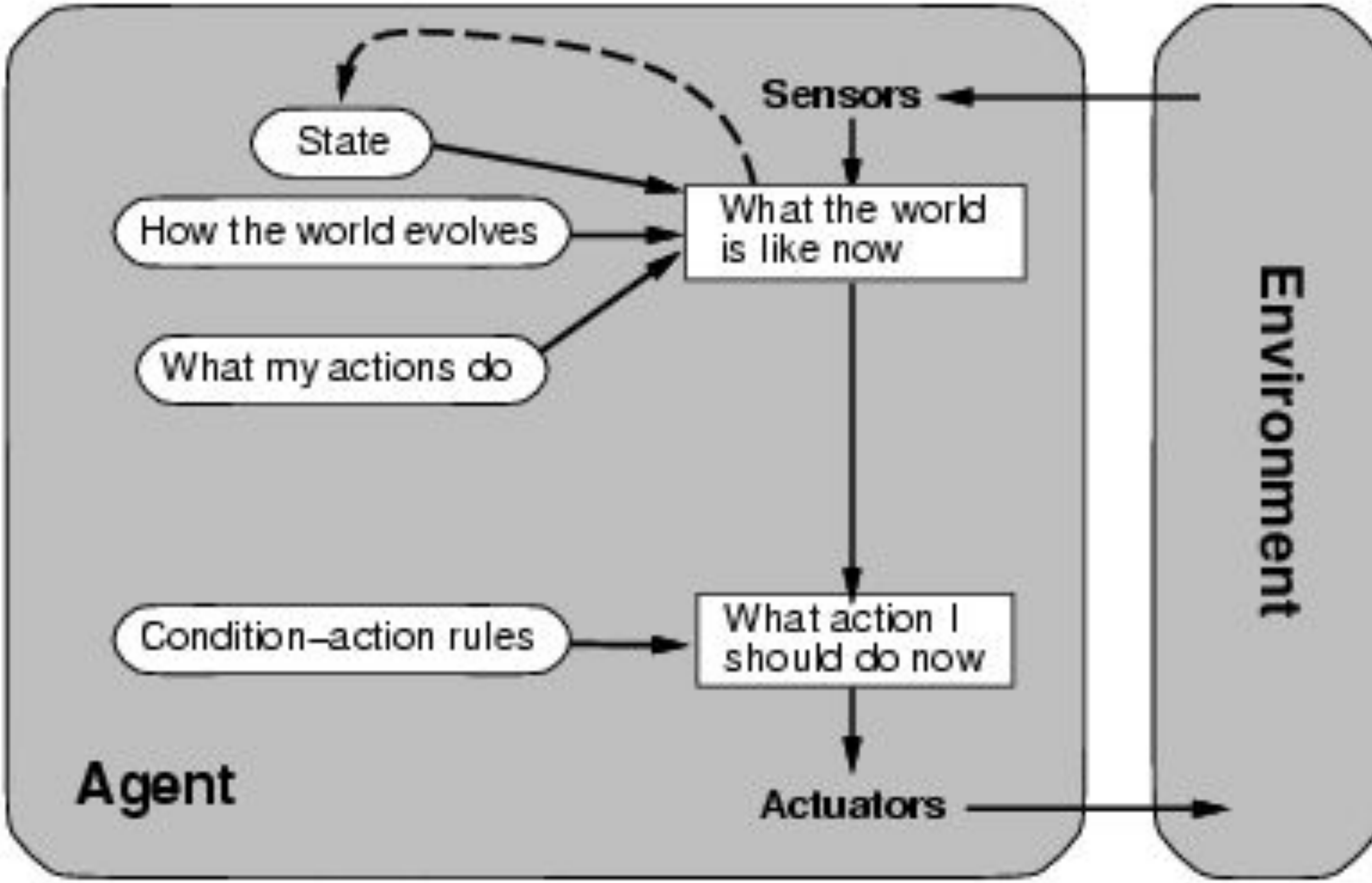
Action becomes a "reflex."

Uses **condition-action rules**.

Agent selects actions on the basis of *current* percept only.



Model-based reflex agents



Model-based reflex agents

Key difference (wrt simple reflex agents):

- Agents have **internal state**, which is used to keep track of past states of the world.

- Agents have the ability **to represent change in the World**.

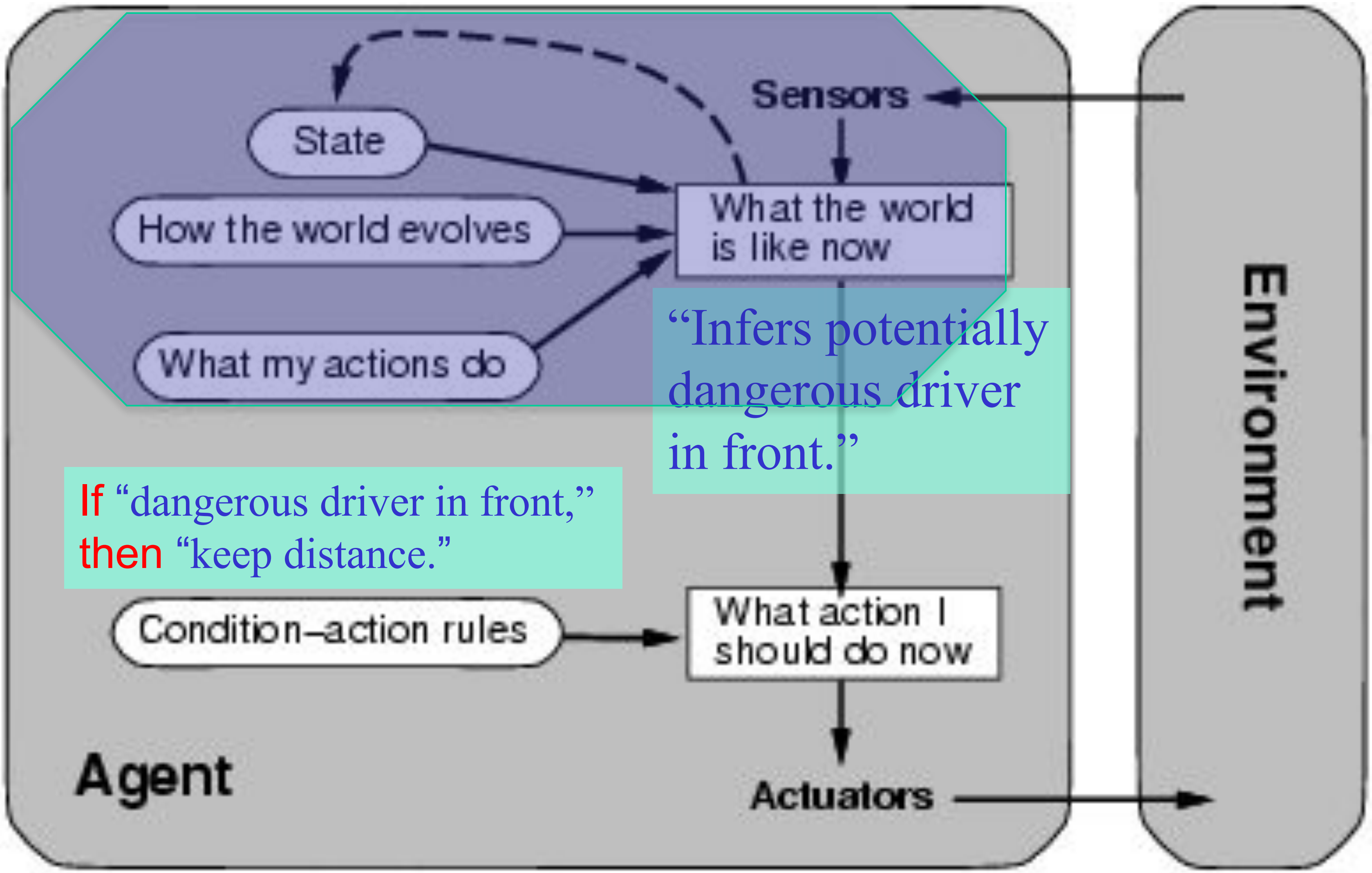
Example: Rodney Brooks' Subsumption Architecture

--- behavior based robots.

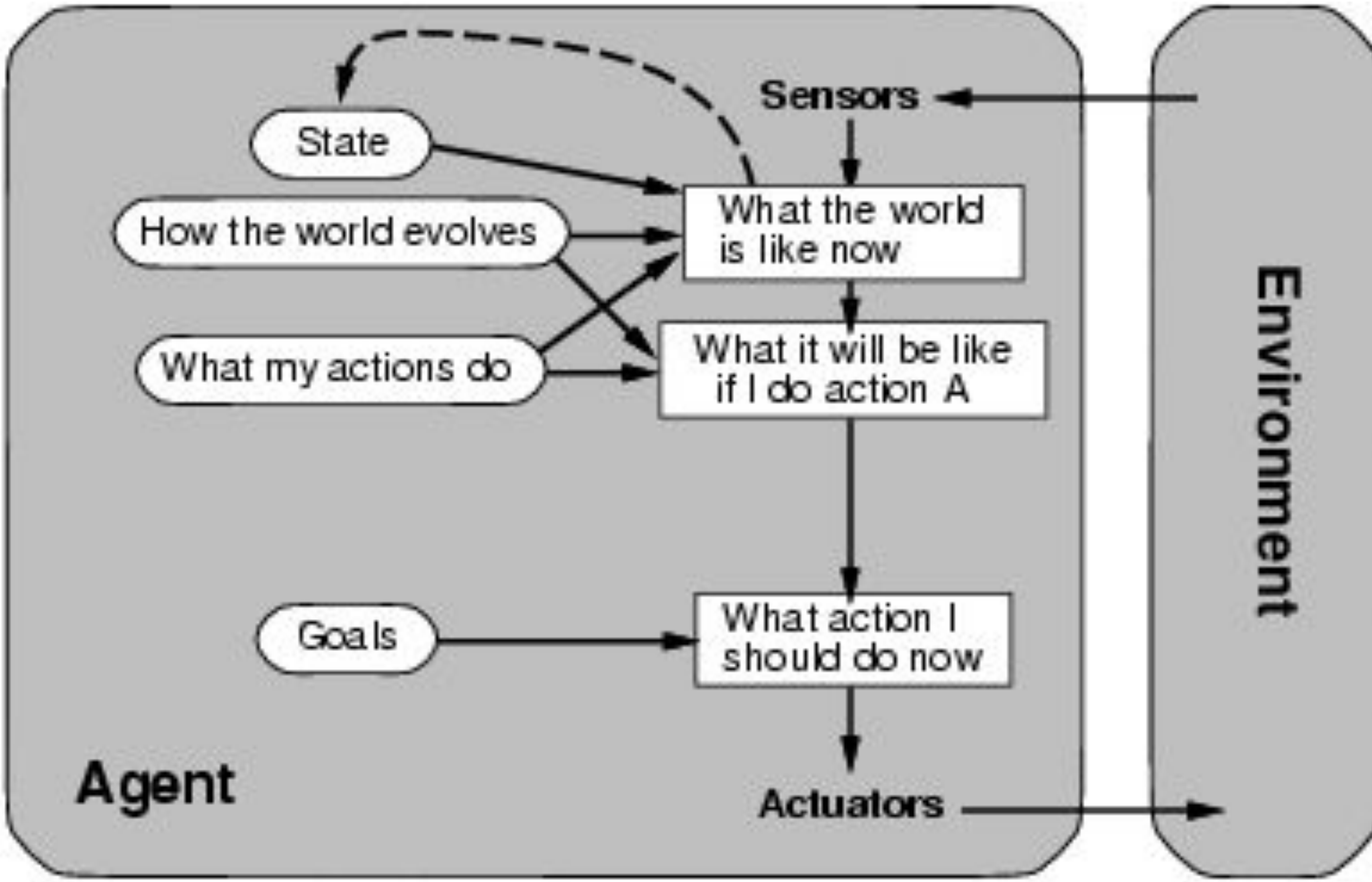
[Here]

Model-based reflex agents

How detailed?



Goal-based agents



Goal-based agents

Key difference wrt Model-Based Agents:

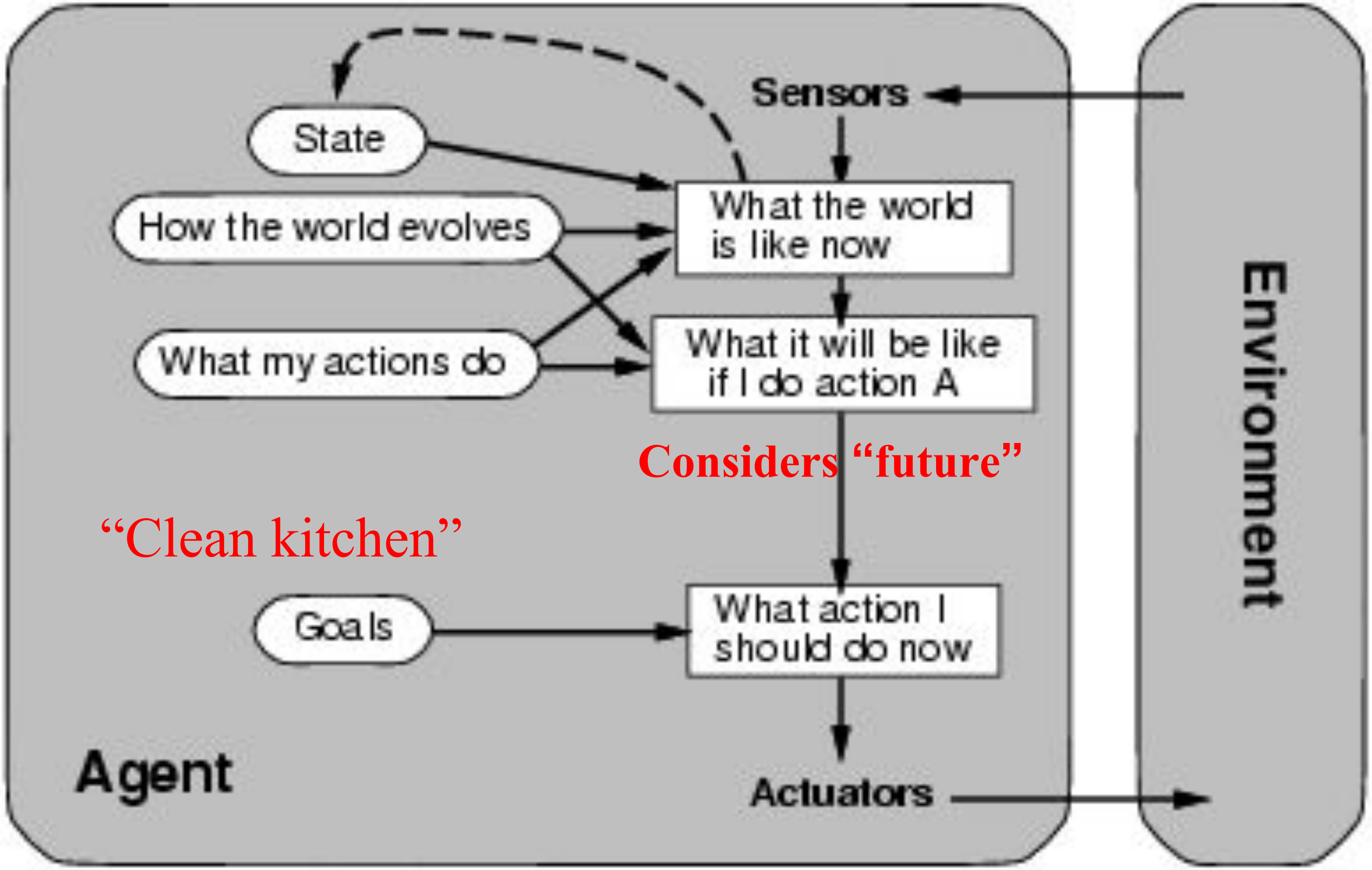
In addition to state information, have **goal information** that **describes desirable situations to be achieved**.

Agents of this kind take future events into consideration.

What *sequence* of actions can I take to achieve certain goals?

Choose actions so as to (eventually) achieve a (given or computed) goal.

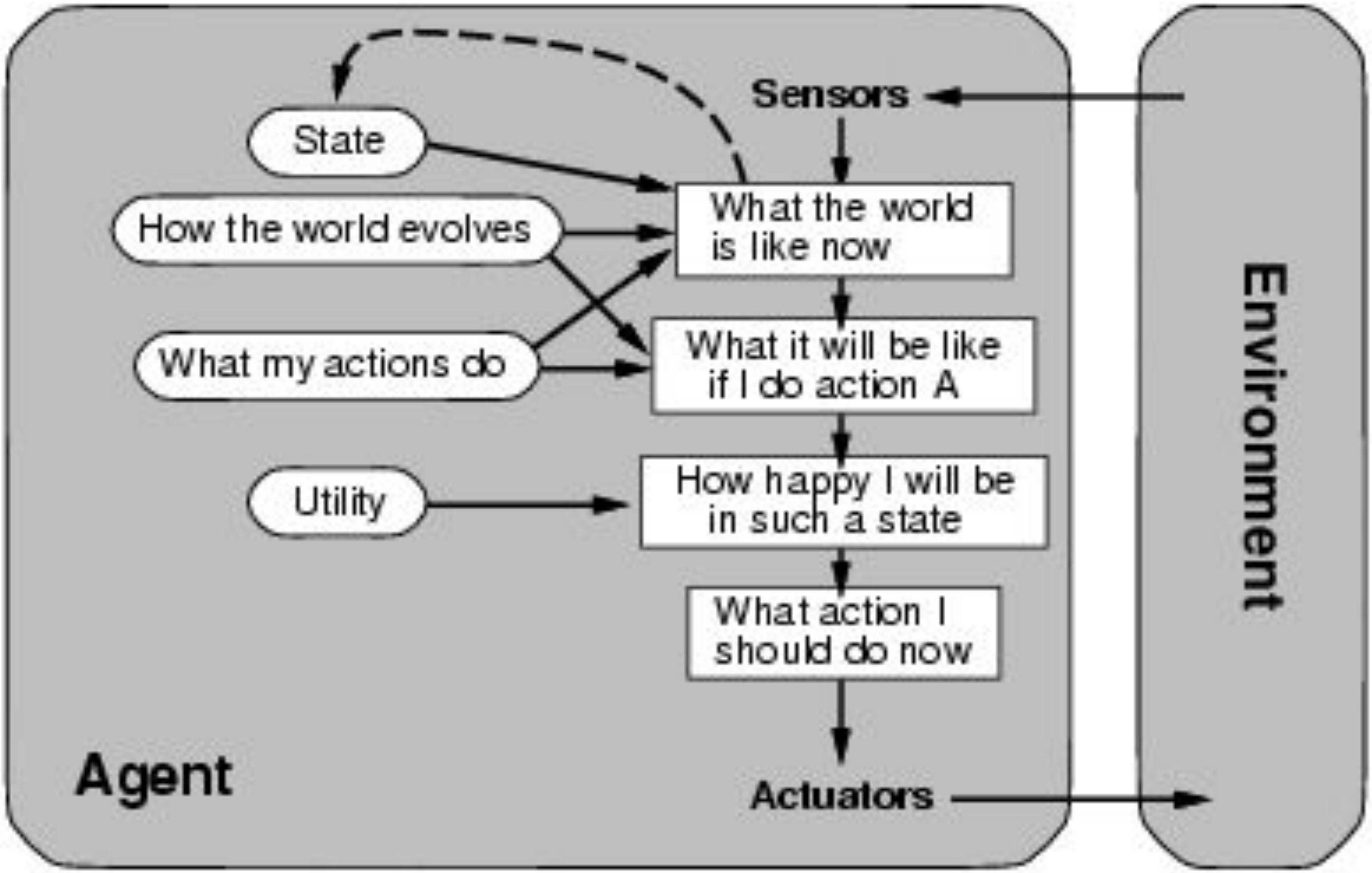
Goal-based agents



Agent keeps track of the world state as well as set of goals it's trying to achieve: chooses actions that will (eventually) lead to the goal(s).

More flexible than reflex agents may involve search and planning

Utility-based agents



Utility-based agents

When there are multiple possible alternatives, how to decide which one is best?

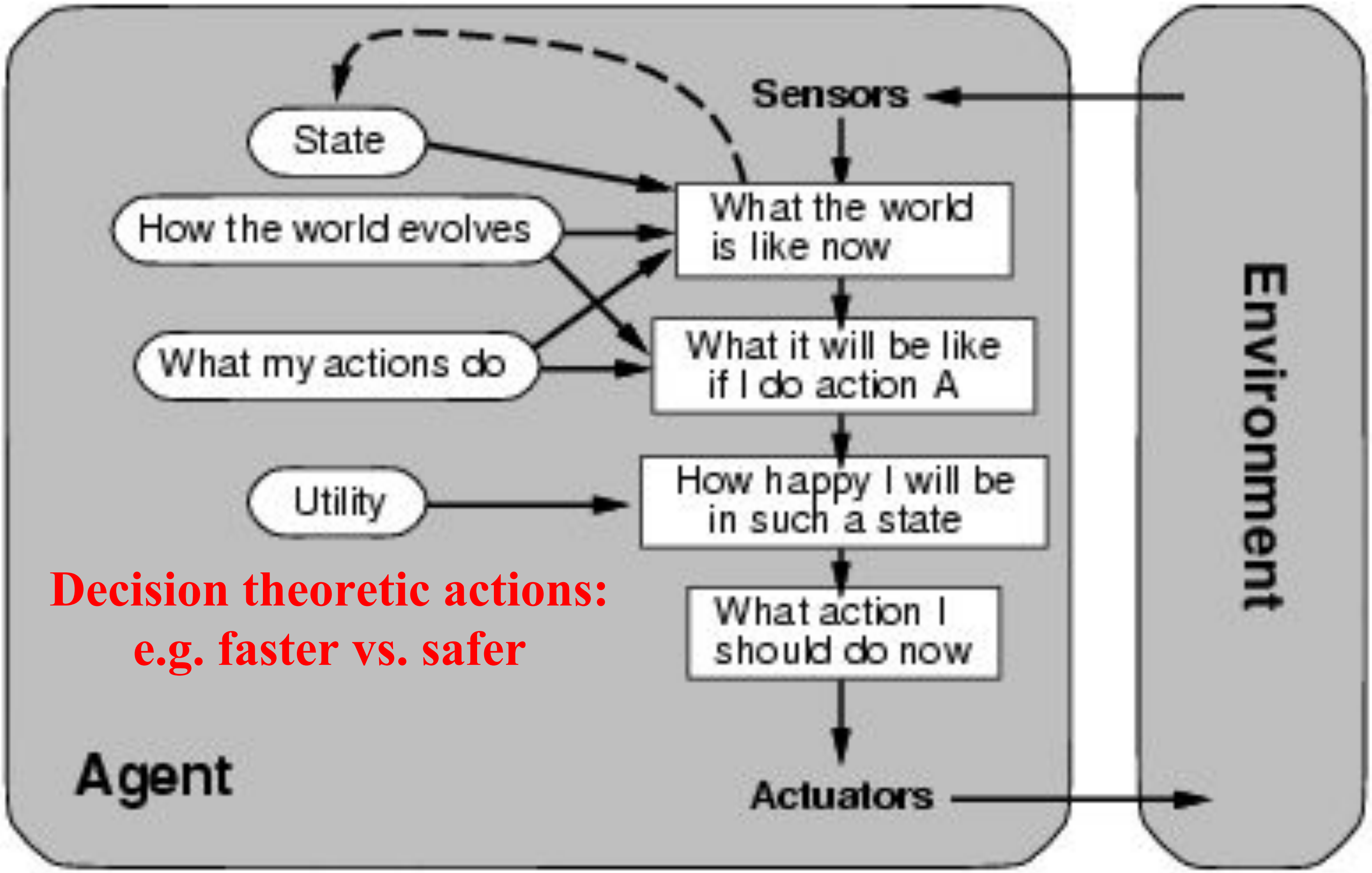
Goals are qualitative: A goal specifies a crude distinction between a happy and unhappy state, but often need a more general performance measure that describes "degree of happiness."

Utility function U: State → R indicating a measure of success or happiness when at a given state.

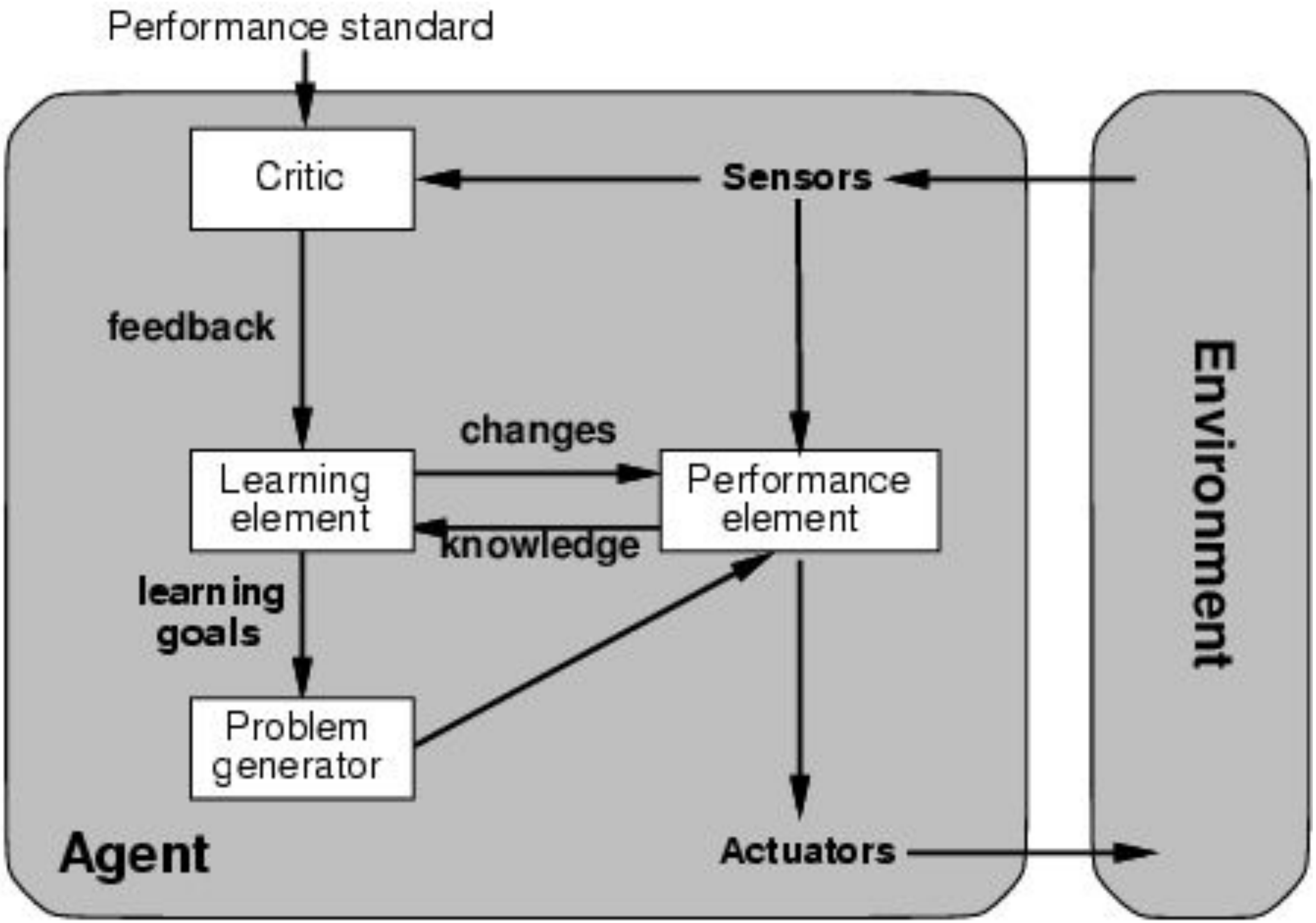
Important for making tradeoffs: Allows decisions comparing choice between conflicting goals, and choice between likelihood of success and importance of goal (if achievement is uncertain).

Use decision theoretic models: e.g., faster vs. safer.

Utility-based agents



Learning agents



Learning agents
Adapt and improve over time

More complicated when agent needs to learn utility information: Reinforcement learning (based on action payoff)