Artificial Intelligence

Lecture #9

Types of Agents

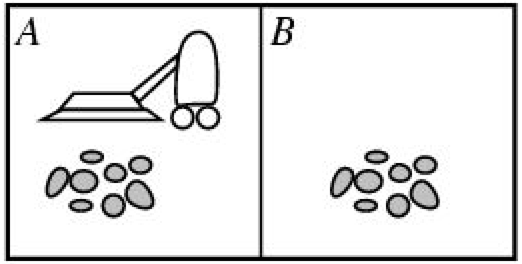
I) --- Table-lookup driven agents

Uses a percept sequence / action table in memory to find the next action. Implemented as a (large) lookup table.

Drawbacks:

- Huge table (often simply too large)
- Takes a long time to build/learn the table

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.

II) --- Simple reflex agents

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

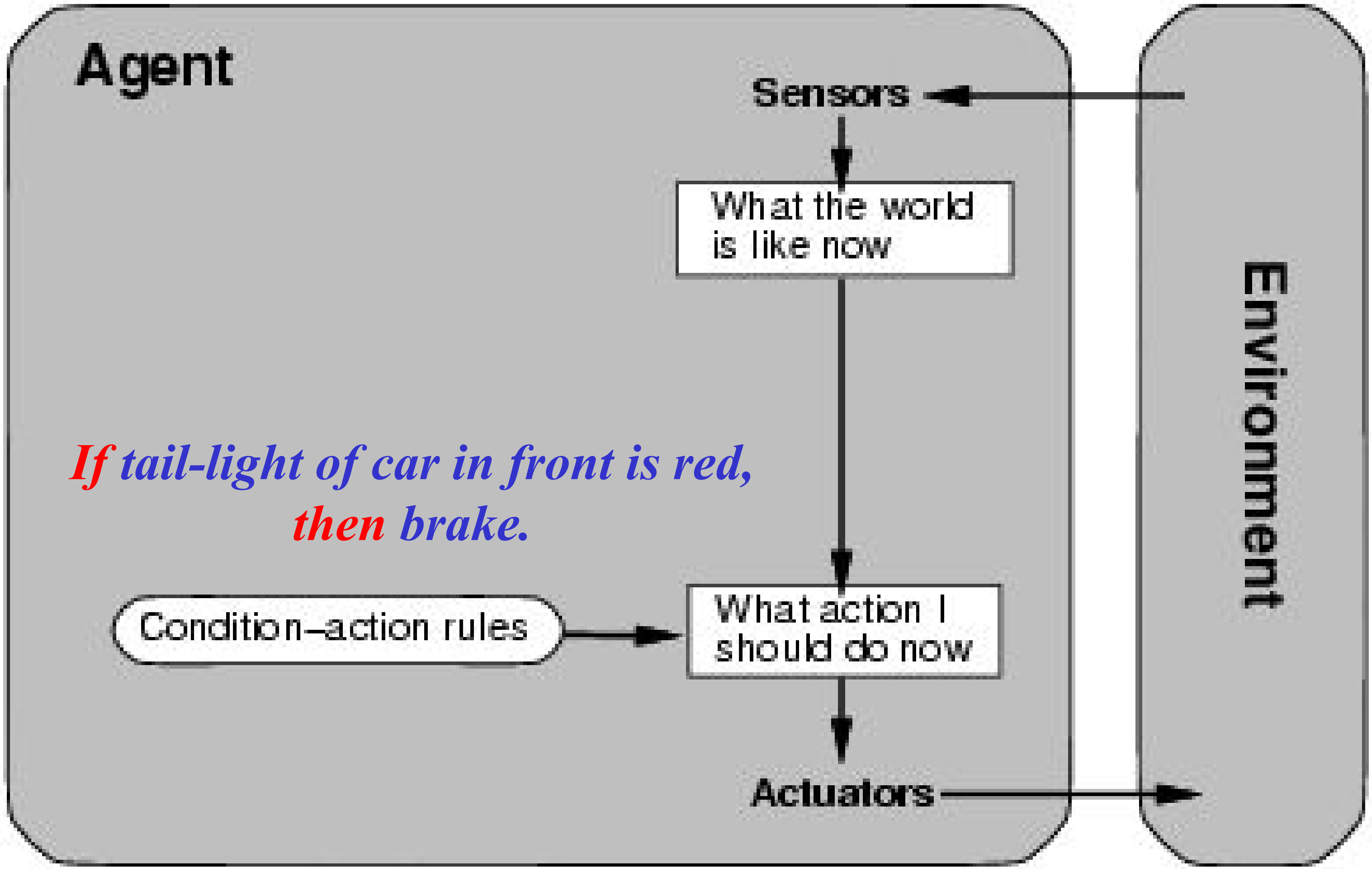
So, actions depend solely on **current percept**.

Action becomes a “reflex.”

Uses **condition-action rules**.

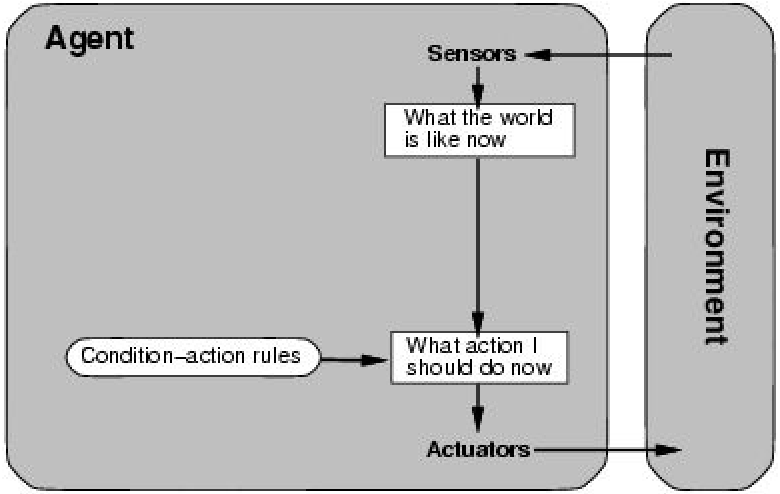
II) --- Simple reflex agents

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



If tail-light of car in front is red, then brake.

II) --- Simple reflex agents



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*

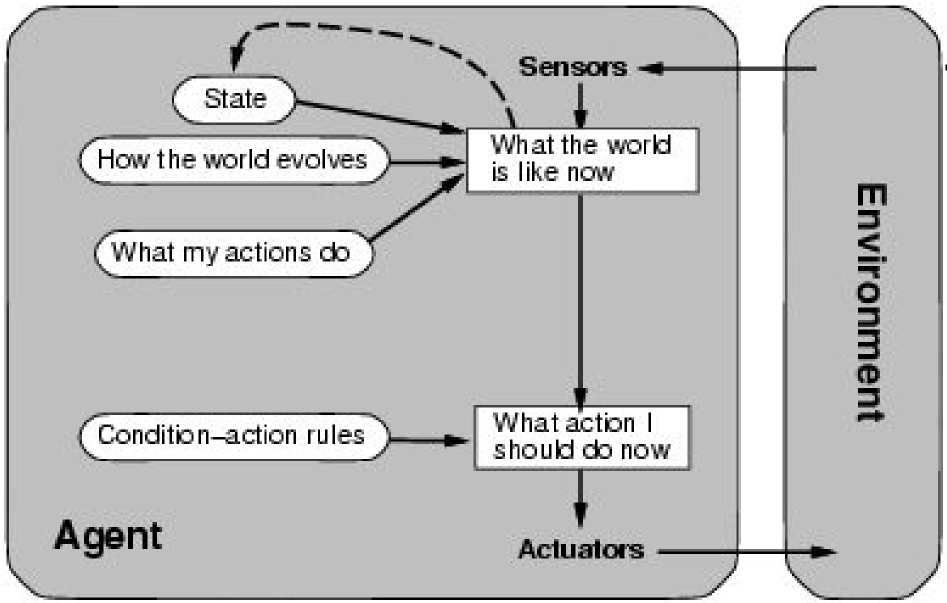
III) --- Model-based reflex agents

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**.

III) --- Model-based reflex agents



- Know how world evolves
  - Overtaking car gets closer from behind
- How agents actions affect the world
  - Wheel turned clockwise takes you right
- Model base agents update

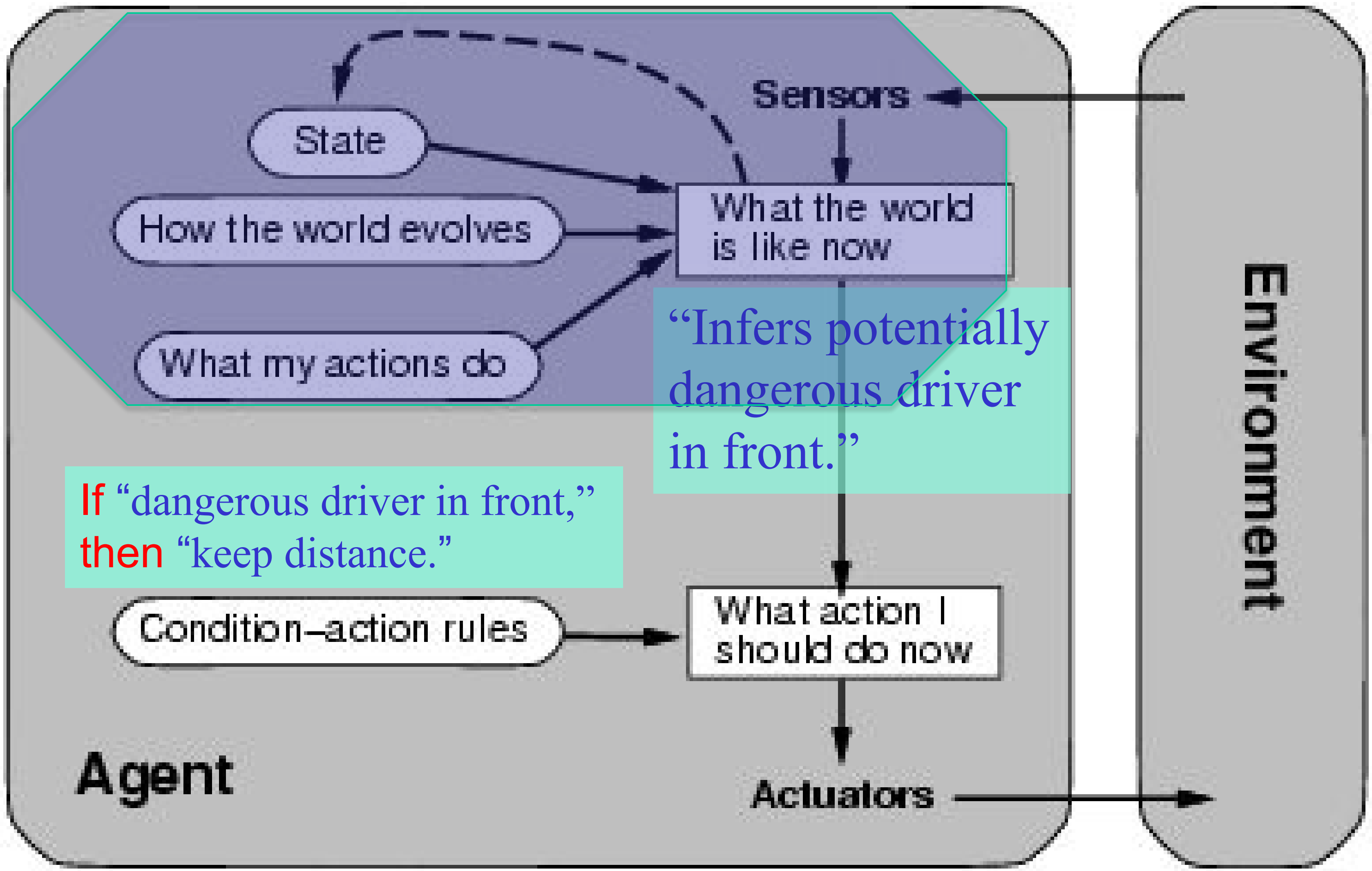
```
function REFLEX-AGENT-WITH-STATE(percept) returns action
   static: state, a description of the current world state
           rules, a set of condition-action rules

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

[Here]

Module: Logical Agents Representation and Reasoning: Part III/IV R&N

Model-based reflex agents

How detailed?



Model-based Reflex Agents

- Deal with partially observable environment
  - an *internal state* maintains important information from previous percepts
  - sensors only provide a partial picture of the environment
  - The *internal states* reflects the agent's knowledge about the world this knowledge is called a *model*.

Model-based Reflex Agents

function REFLEX-AGENT-WITH-STATE (*percept*) **returns** action

static: *state*, a description of the current world state

rules, a set of condition-action rules

action, the most recent action, initially none

state ← UPDATE-STATE (*state, percept*)

rule ← RULE-MATCH (*state, rules*)

action ← RULE-ACTION [*rule*]

return *action*

- TO Update internal state information , agent must know
  - How the world evolves independently of the agent
  - How the agents own actions affect the world

Model-Based Vs. Simple Reflex

Example - Taxi Driver changing position

Model-based

- percept -no car
- internal state -to keep track where the other cars are
- update state-
  - overtaking car will be closer behind
  - whether he should turns the steering wheel clockwise or anti...

Simple Reflex

- percept -no car
- action - just change his position

Model-based Reflex Agents

- *Advantages*
  - Sensors do not provide access to the complete state of the world
  - Internal state helps agent to distinguish between world states
- *Disadvantages*
  - More complex than simple reflex agent
  - Computation time increases

IV) --- 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.

→ ***problem solving and search! (R&N --- Part II, chapters 3 to 6)***

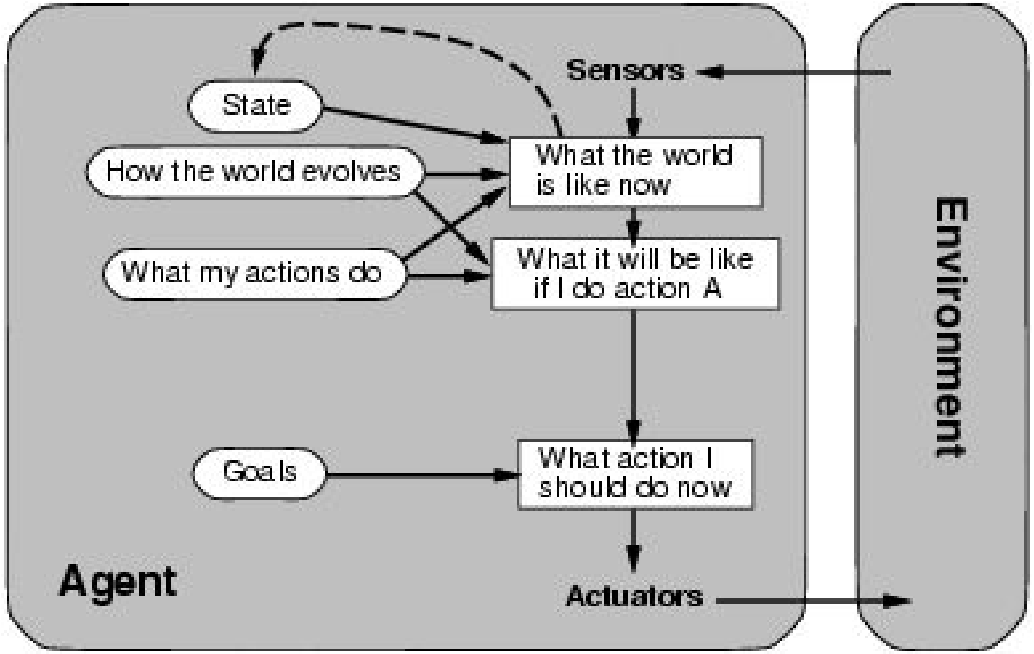
Goal-based agents

- knowing state and environment? Enough?
  - Taxi can go left, right, straight
- Have a goal
  - A destination to get to

Uses knowledge about a goal to guide its actions

- E.g., Search, planning

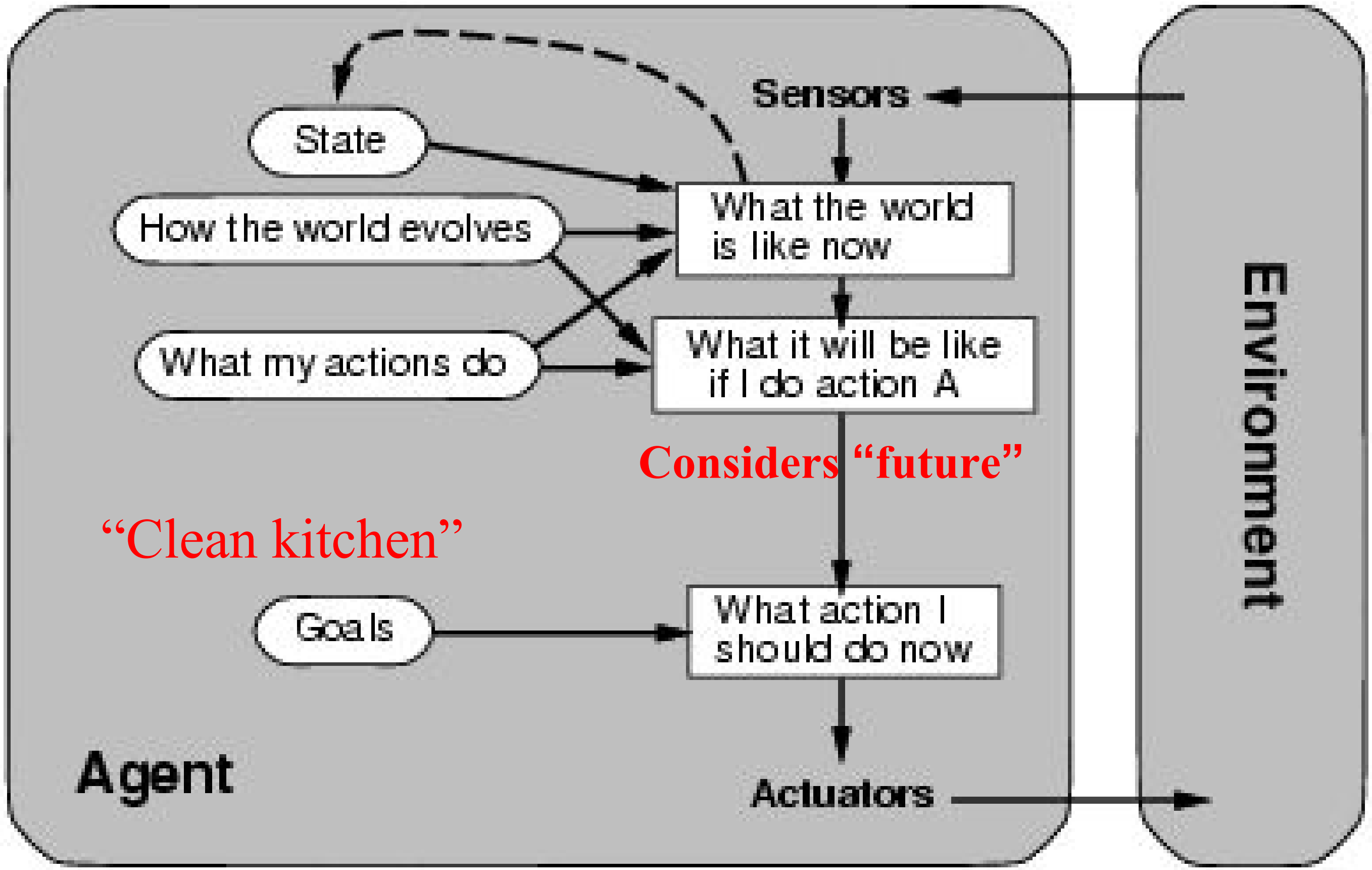
Goal-based agents



- Reflex agent breaks when it sees brake lights. Goal based agent reasons
  - Brake light -> car in front is stopping -> I should stop -> I should use brake

Module:
Problem Solving

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

V) --- 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 $\rightarrow$ 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

Goals are not always enough

- Many action sequences get taxi to destination
- Consider other things. How fast, how safe…..

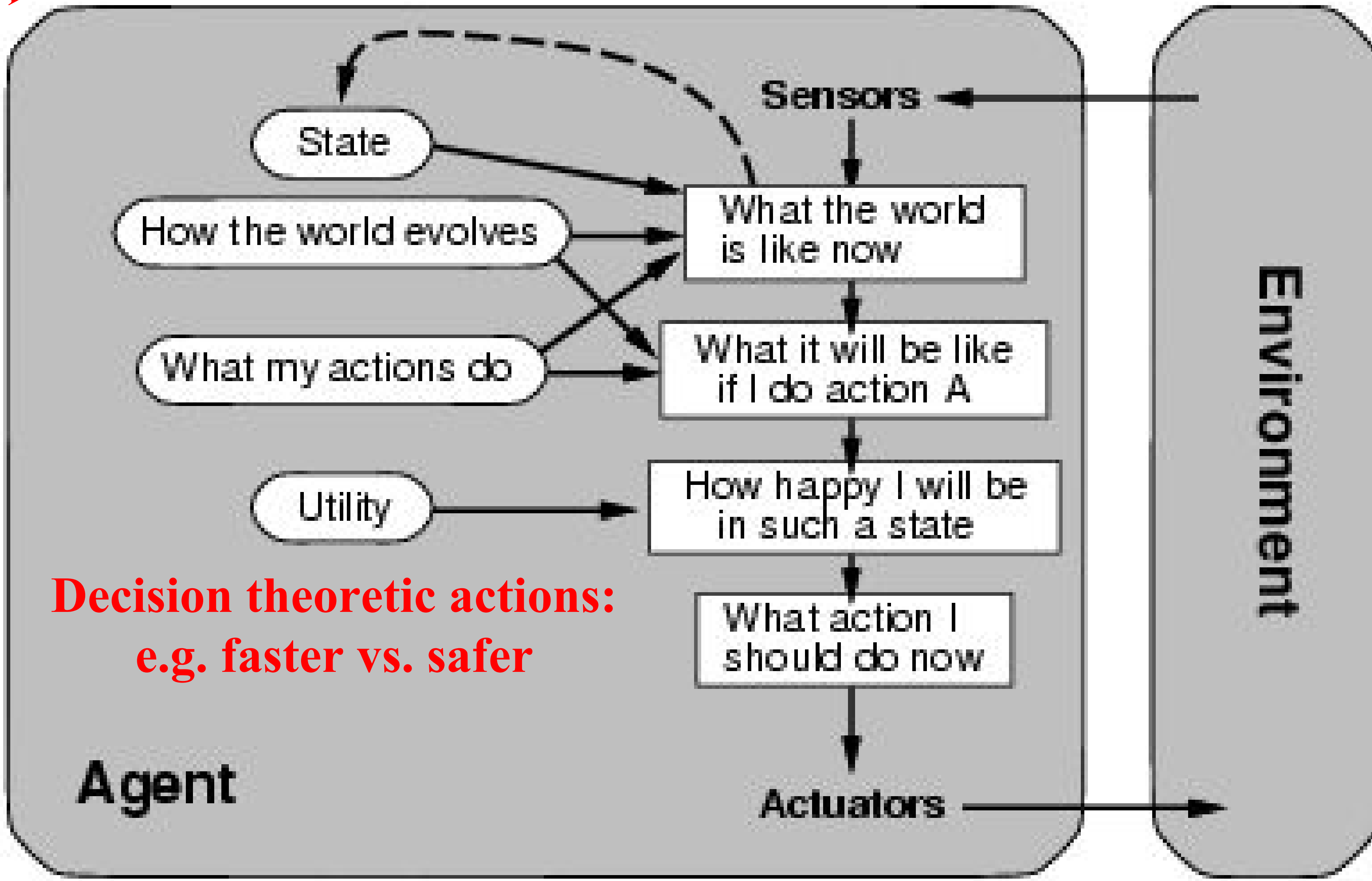
A utility function maps a state onto a real number which describes the associated degree of “happiness”, “goodness”, “success”.

Where does the utility measure come from?

- Economics: money.
- Biology: number of offspring.
- Your life?

Module: Decision Making

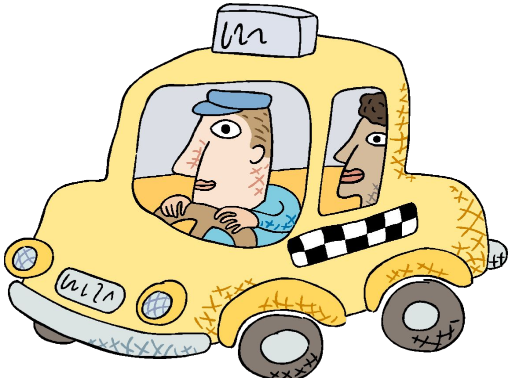
Utility-based agents



Decision theoretic actions:
e.g. faster vs. safer

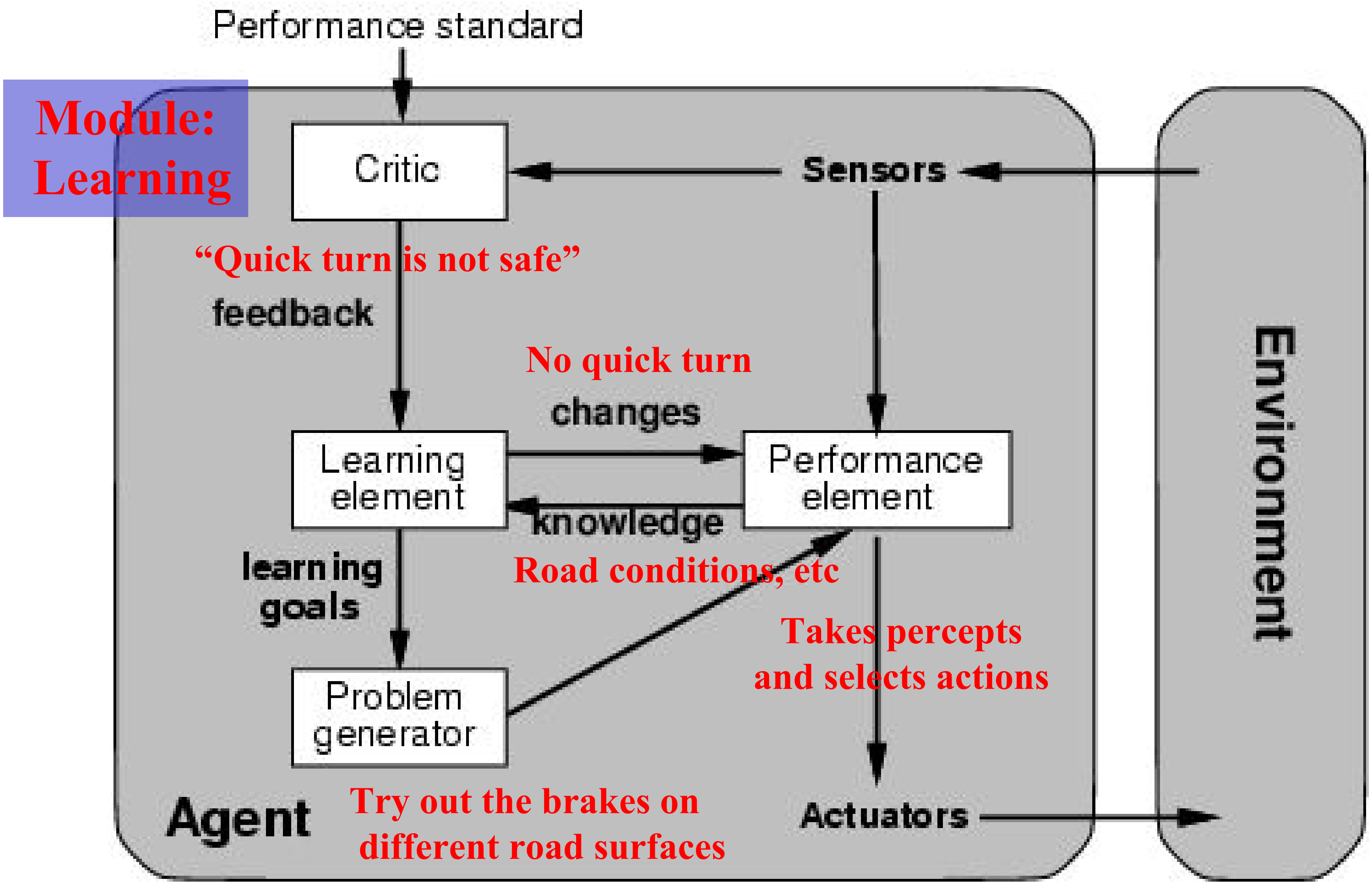
Utility-based Agents

- *Advantage:*
  - Utility based agents can handle the uncertainty inherent in partially observable environments.
- *Consider the taxi driver example:*
  - There are many action sequences that will get the taxi to its destination but some are quicker ,safer, more reliable or cheaper than others.
  - Goals just provides a distinction between whether the passenger is ‘happy’ or ‘unhappy’. The utility function defines the degree of happiness.



VI) --- Learning agents
Adapt and improve over time

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



Learning Agents

- The *Learning Element* is responsible for making improvements.
  - It takes some knowledge and some feedback on how the agent is doing, and determines how the Performance Element should be modified to do better in the future.
  - It is also responsible for improving the *efficiency* of the Performance Element.

Learning Agents

- The *Performance Element* is responsible for selecting external actions.
  - It is what we have previously considered to be the entire agent--- it takes in percepts and decides on actions.
  - The design of the Learning Element depends very much on the design of the Performance Element.

Learning Agents

- The *Critic* is designed to tell the Learning Element how well the agent is doing.
  - It employs a fixed *standard of performance*.
  - This is necessary because the percepts themselves provide no indication of the agent’s success.

Learning Agents

- The *Problem Generator* is responsible for *suggesting actions* that will lead to new and informative experience.
  - The point is that if the Performance Element had its way, it would keep doing the actions that are best, given what it knows.
  - But if the agent is willing to explore a little, and do some (perhaps) sub-optimal actions in the short run, it might discover much better actions for the long run. T
  - The Problem Generator's job is to suggest these *exploratory actions.*

Learning Agents

The Taxi-Driver Example

- The Performance Element consists of whatever collection of knowledge and procedures the taxi has for selecting its driving actions(turning, accelerating, braking, and so on). The taxi goes out on the road and drives, using this performance element.
- The Learning Element formulates goals, for example, to learn better rules describing the effects of braking and accelerating, to learn the geography of the area, to learn how the taxi behaves on wet roads, and to learn what causes annoyance to other drivers.

Learning Agents

The Taxi-Driver Example

- The Critic observes the world and passes information along to the learning element. For ex: after the taxi makes a quick left turn across the three lanes of traffic, the critic observes the "shocking" language used by other drivers, the learning element is able to formulate a rule saying this is a bad action, and the performance element is modified by installing the new rule.
- The Problem Generator kicks in with a new suggestion; try taking 7th avenue
- uptown this time, and see if it is faster than the normal route.

Summary: agent types

(1) Table-driven agents

– use a percept sequence/action table in memory to find the next action. They are implemented by a (large) lookup table.

(2) Simple reflex agents

– are based on condition-action rules, implemented with an appropriate production system. They are <u>stateless devices which do not have memory of past world states.</u>

(3) Agents with memory - Model-based reflex agents

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

(4) Agents with goals – Goal-based agents

– are agents that, in addition to state information, have goal information that describes desirable situations. Agents of this kind take future events into consideration.

(5) Utility-based agents

– base their decisions on classic axiomatic utility theory in order to act rationally.

(6) Learning agents

– they have the ability to improve performance through learning.

Summary

An agent perceives and acts in an environment, has an architecture, and is implemented by an agent program.

A rational agent always chooses the action which maximizes its expected performance, given its percept sequence so far.

An autonomous agent uses its own experience rather than built-in knowledge of the environment by the designer.

An agent program maps from percept to action and updates its internal state.

- Reflex agents (simple / model-based) respond immediately to percepts.
- Goal-based agents act in order to achieve their goal(s), possible sequence of steps.
- Utility-based agents maximize their own utility function.
- Learning agents improve their performance through learning.

Representing knowledge is important for successful agent design.

The most challenging environments are partially observable, stochastic, sequential, dynamic, and continuous, and contain multiple intelligent agents.