331 – Intro to Intelligent Systems
Week01c
Intelligent Agents
R&N Chapter 2

T.J. Borrelli

Agents and Environments

- Last time we spoke about **rational agents** and their central role to AI
- Now we define this more formally with the goal of identifying design principles for building successful agents
- In this context “successful” means a system that can reasonably be thought of an intelligent
- A rational agent should behave “as well as possible” given the environment/conditions it must exist within
- **Agent –** anything that perceives environment through sensors and acts upon environment through actuators

Agents

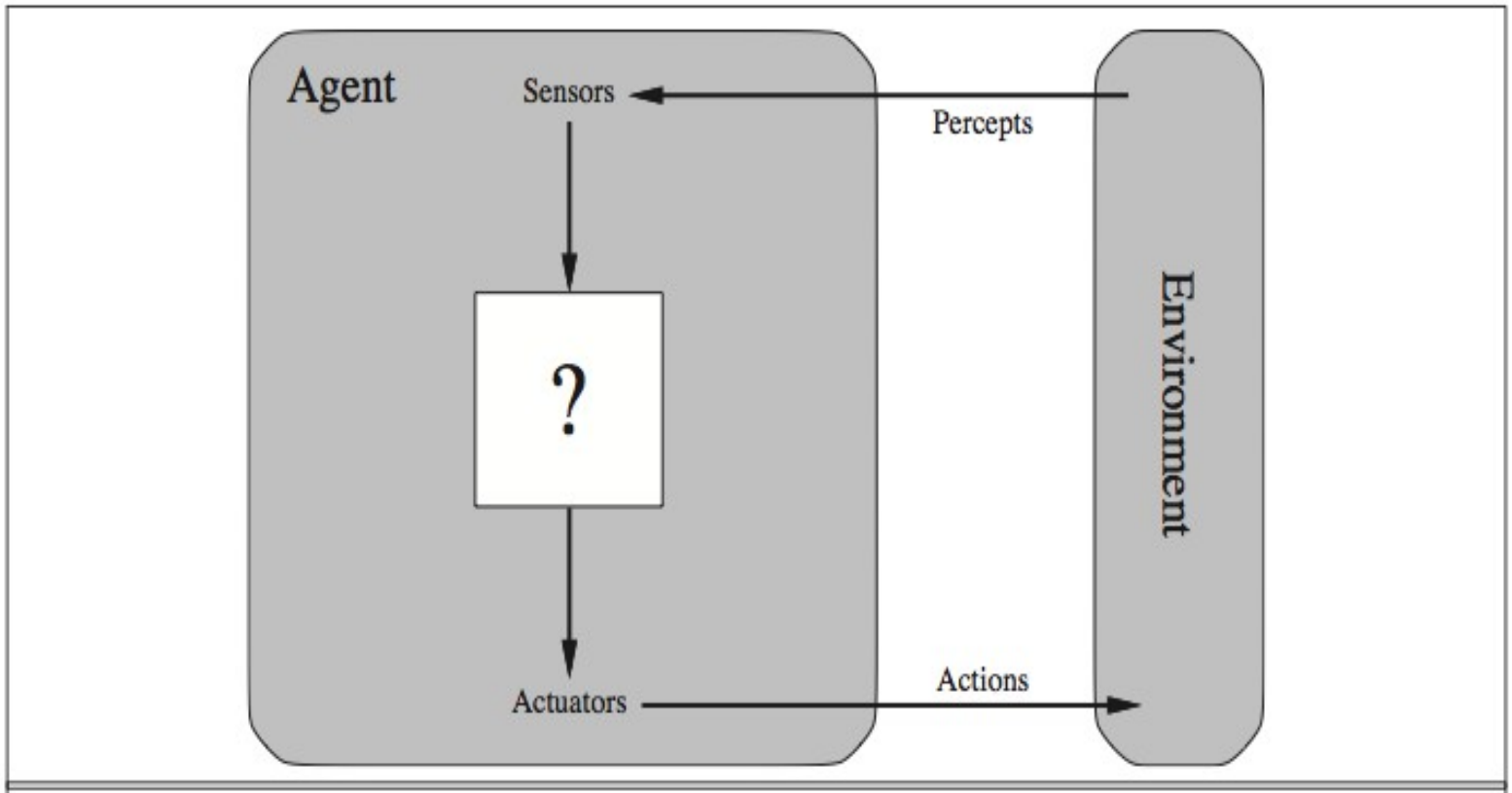


Figure 2.1 FILES: figures/agent-environment.eps (Tue Nov 3 16:22:19 2009). Agents interact with environments through sensors and actuators.

Rational agents

- Rational agents do the “right” thing
- How do we define this?
- In terms of the consequences of a given action
- Performance measure – design these according to the outcome we want in the environment

Rationality

- What is rational at any given time depends on
  - The performance measure (defines success)
  - The agent's prior knowledge of environment
  - The actions that the agent can perform
  - The agent's percept sequence to date (history of things perceived)
- We can now define a **rational agent:** For each possible percept sequence (input), a rational agent should select an action to maximize its performance measure given it percept sequence and other knowledge

Rationality

- Rationality is not the same as perfection
- It is typically not possible to be able to perfectly predict the actual outcomes of every action (that would be omniscience).
- Rationality therefore maximizes *expected* performance

Rational Agents and learning

- Rational agents can often engage in information gathers – performing certain actions in order to modify future percepts
- Exploration is an example of this
- Exploration is useful in environments that are unknown initially
- Our rational agents should also **learn** about their environment
- The initial configuration could reflect some prior knowledge or base rules and we can add to this by exploration

PEAS and the nature of environments

- How do we now build rational agents?
- Need to think about **task environments –** the “problems” our agents are “solutions” for
- Our Task Environment covers
- **P – P**erformance measure (often will involve trade-offs)
- **E – E**nvironment
- **A – A**ctuators
- **S – S**ensors

PEAS example

- Example of an automated taxi driver (open-ended)

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits,	Roads, other traffic, pedestrians, customers	Steering, acceleration, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors

Properties of task environments

- Fully observable vs. partially observable
- Single agent vs. multiagent
- Deterministic vs. stochastic
- Episodic vs. sequential
- Static vs. dynamic
- Discrete vs. continuous
- Known vs. unknown

Fully observable vs. Partially observable

- Fully observable: If the agent's sensors give it access to the complete state of the *relevant* information from the environment at each point in time
- Nice because we don't need to keep track of lots of state information
- Partially observable when we get a subset of information from environment
- Also possible that the environment is unobservable – can still sometimes achieve goals

Single agent vs. multiagent

- Are we working against other agents (competition)
- Or are we working in conjunction with other agents (cooperation)
- Are we the only agent engaged in the environment

Deterministic vs. stochastic

- **Deterministic** – If the next state of the environment is completely determined by the current state and the action executed by the agent
- **Stochastic** – otherwise

- An environment is **uncertain** if it is not fully observable or not deterministic

Episodic vs. Sequential

- **Episodic** – agent's experience is divided into atomic (indivisible) parts; in each part agent receives a percept and performs a single action; next episode does not depend on actions taken in previous

- **Sequential** – The current decision could affect all future decisions (e.g. chess and taxi driving)

Static vs. Dynamic

- **Static** – environment does not change while agent is deciding on next action (e.g. crossword puzzles)
- **Dynamic** – environment may change while agent is deliberating (e.g. taxi driving)
- **Semi-dynamic** – environment does not change but the agent's performance score does

Discrete vs. Continuous

- **Discrete** – state of environment and time are separate and distinct slices
- For example, Chess has distinct set of percepts and actions

- **Continuous** –state of environment and time are along a spectrum of possible values (e.g. taxi driving)

Known vs. Unknown

- **Known** – agent state of knowledge about the rules/laws of physics in the environment
- The outcomes (or probabilities of outcomes) are known for all actions
- **Unknown** – If environment is unknown agent will have to learn how it works to make good decisions
- Note that the distinction between known and unknown environments is not the same as observable and partially observable
- It is possible for known environments to be partially observable (solitaire – know the rules, but can't see all the cards)
- Converse is also true, possible for an unknown environment to be fully observable (a new video game – screen shows all elements but don't know what buttons to press until you try them)

Tough combination

- Hardest case of the above is partly observable (or unobservable), multiagent, stochastic, sequential, dynamic, continuous and unknown
- Our taxi driving agent represents all of these

What is each of the following

	Crossword puzzle	Spam filter	Chess w/ clock	Image analysis	Poker	Interactive tutor
Observable						
Single vs multi						
Deterministic						
Episodic						
Static						
Discrete						
Known						

Simple agent pseudocode

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

persistent: *percepts,* a sequence, initially empty

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

append *percept* to end of *percepts*

action ← LOOKUP(*percepts, table)*

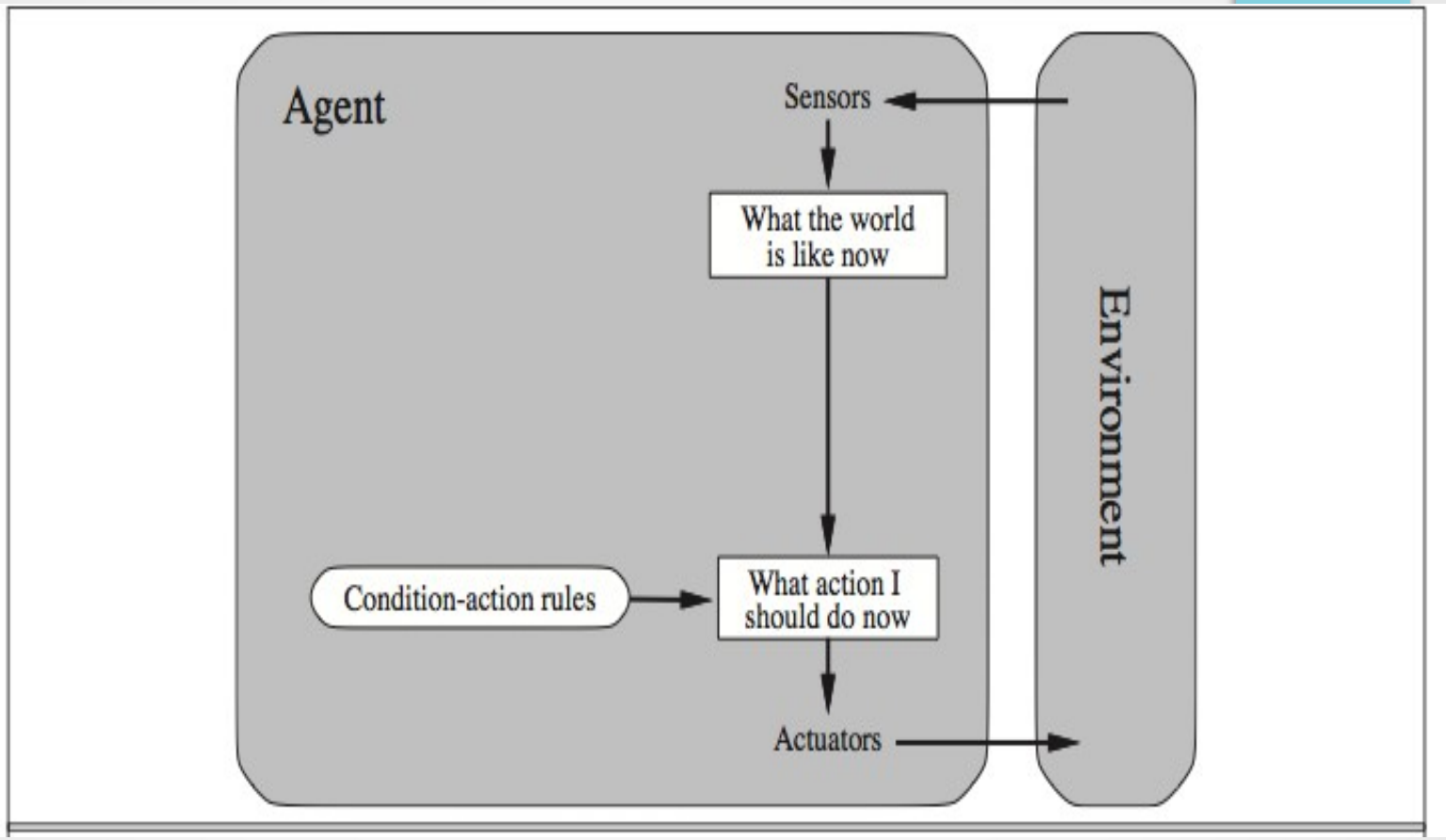
return *action*

- Describes a trivial agent program that keeps track of the percept sequence and then uses it to index into a table of actions
- Table represents the agent function that the agent program embodies
- To build a rational agent – we must construct a table that contains the appropriate action for each percept sequence

Basic Agent Architectures

- Simple reflex
- Reflex with state
- Goal-based
- Utility-based

Simple Reflex Agent



- Responds directly to percepts – acts according to a rule whose condition matches the current state

Simple Reflex Agent pseudocode

function SIMPLE-REFLEX-AGENT(*percept*) **returns** action

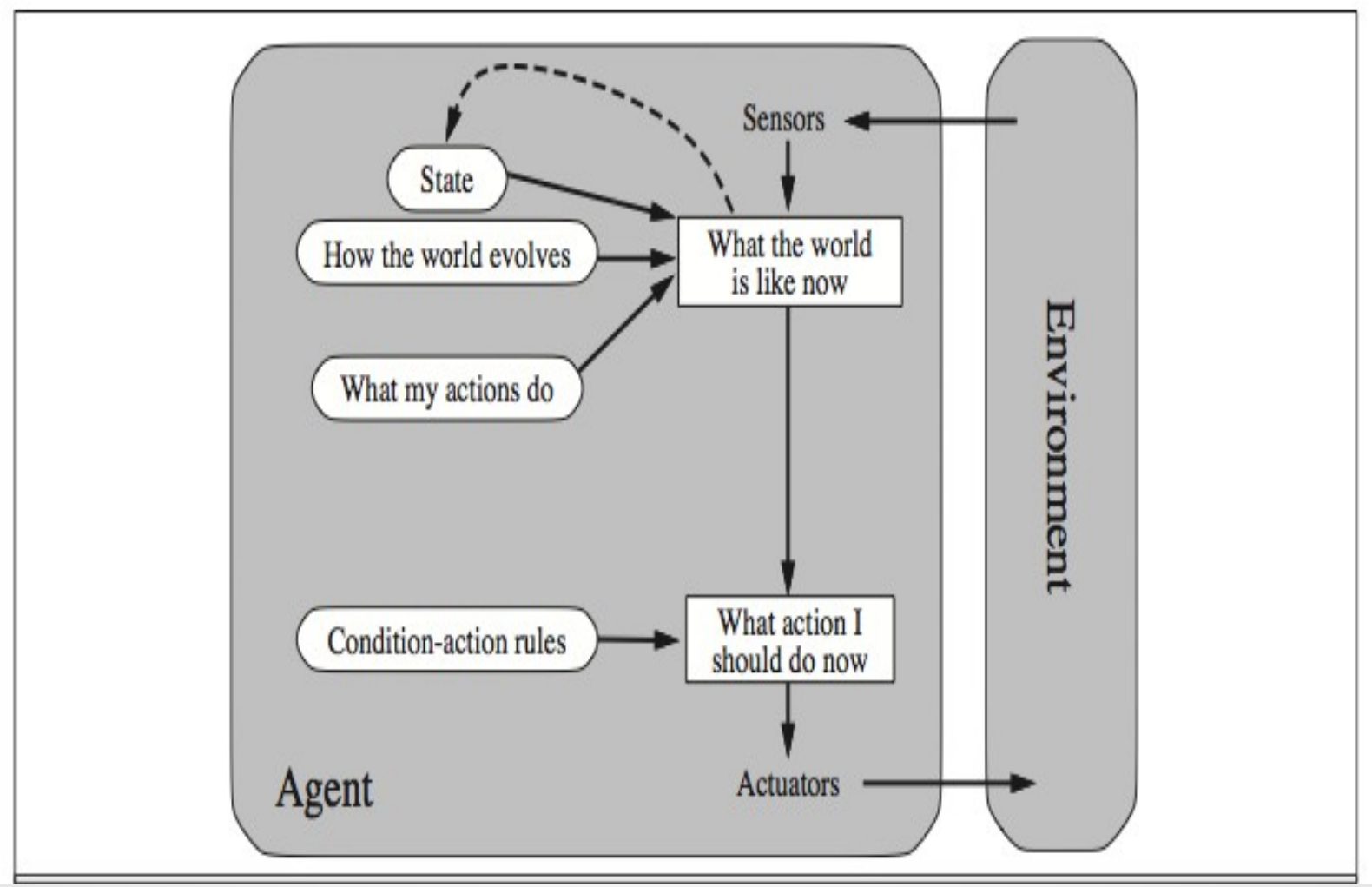
persistent: *rules,* set of condition-action rules

state ← INTERPRET-INPUT(*percepts*)

action ← RULE-MATCH(*state, rules)*

return *action*

Model-based reflex agent



- Maintain internal state to track aspects of world that are not evident in current percept

Model-Based Agent pseudocode

function MODEL-BASED-REFLEX-AGENT(*percept*) **returns** action

persistent: *state,* the agent's current belief of world state

rules, set of condition-action rules

action, the most recent action, initially none

model, desc. how next state depends on cur. state & action

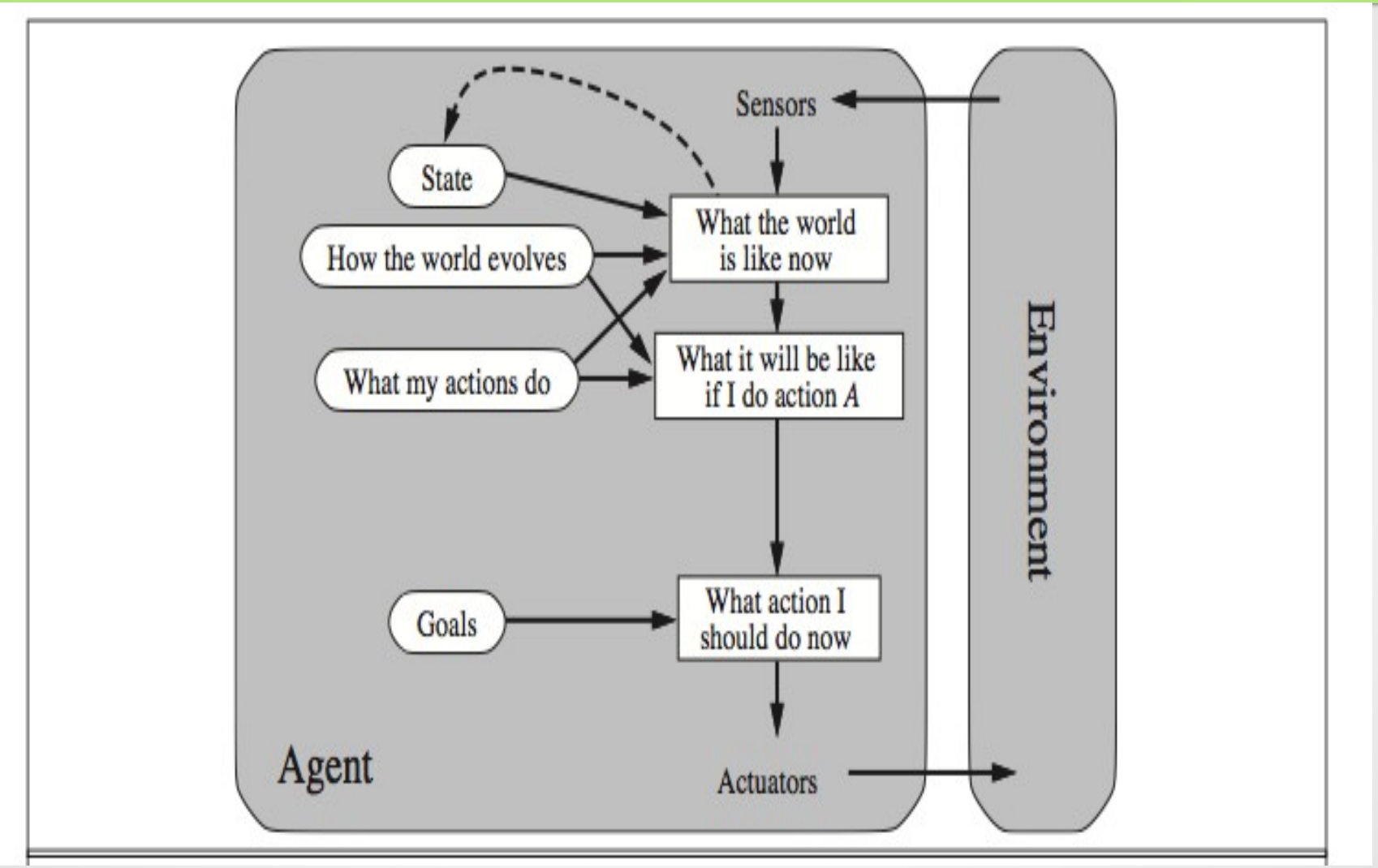
state ← UPDATE-STATE(*state, action, percept, model*)

rule ← RULE-MATCH(*state, rules)*

action ← *rule.*ACTION

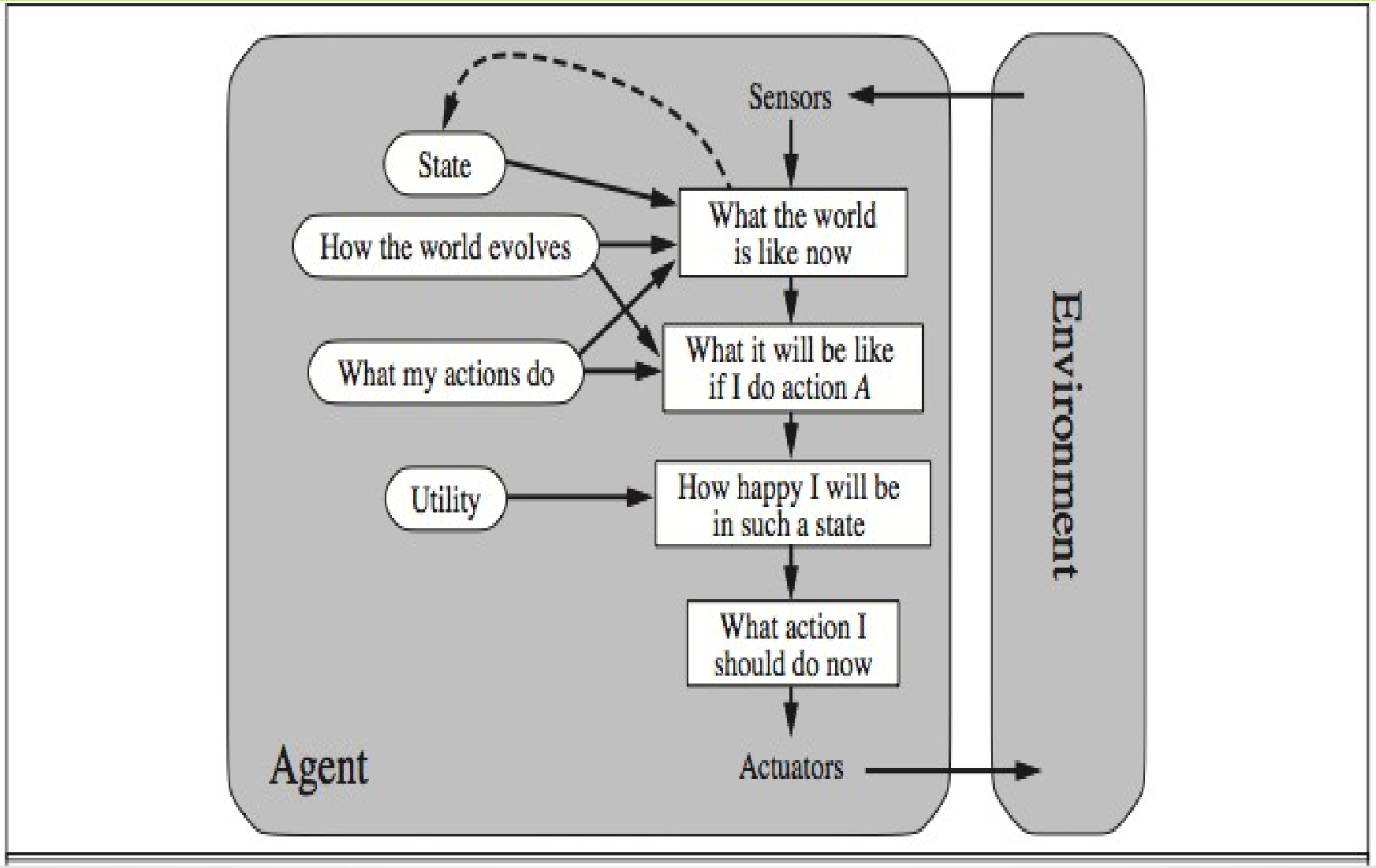
return *action*

Model-based, Goal-based agent



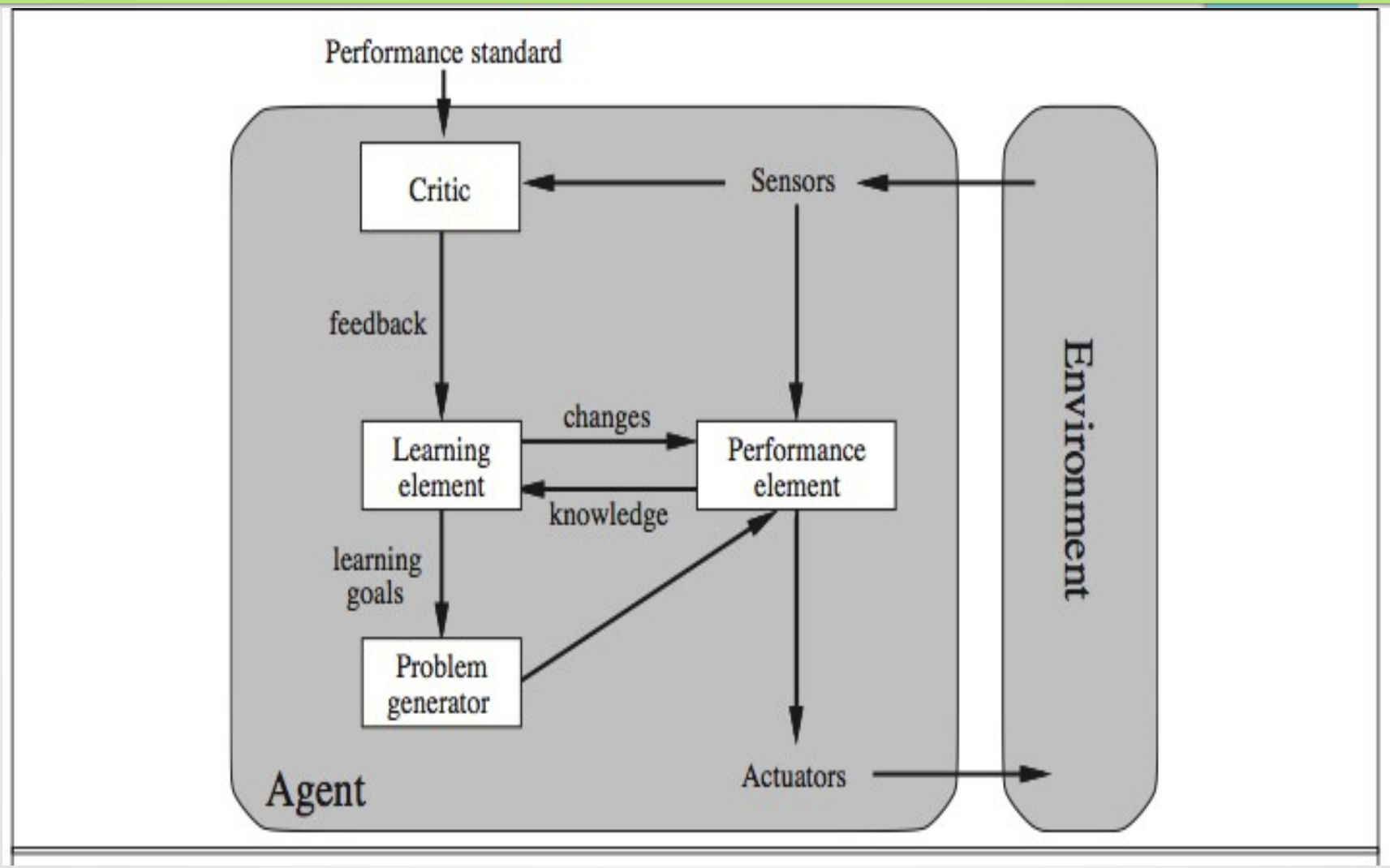
- Acts to achieve certain goals

Utility-based Agent



- Tries to maximize their own expected utility (“happiness”)

General learning agent



- Goal of learning: to improve performance

Summary

- Agents interact with the environment using sensors and actuators
- Performance measure evaluates environment state sequence
- A perfectly rational agent maximizes (expected) performance
- PEAS descriptions define task environments
- Environments categorized along different dimensions
  - Observable, deterministic, episodic, static, discrete, single/multi . . .
- Basic agent architectures
  - Simple reflex
  - Reflex with state
  - Goal-based
  - Utility-based