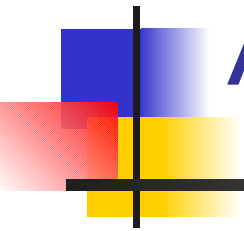


AI and Agents

CS 171/ 271

(Chapters 1 and 2)

Some text and images in these slides were drawn from Russel & Norvig's published material

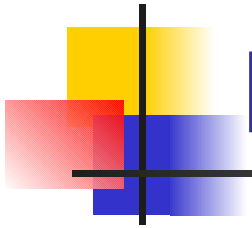
What is Artificial Intelligence?

Definitions of AI vary

Artificial Intelligence is the study of systems that

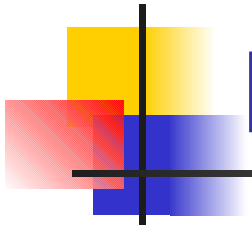
think like humans	think rationally
act like humans	act rationally

Systems Acting like Humans

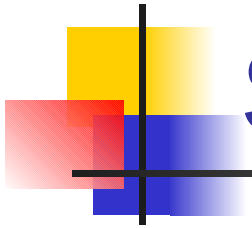


- Turing test: test for intelligent behavior
  - Interrogator writes questions and receives answers
  - System providing the answers passes the test if interrogator cannot tell whether the answers come from a person or not
- Necessary components of such a system form major AI sub-disciplines:
  - Natural language, knowledge representation, automated reasoning, machine learning

Systems Thinking like Humans

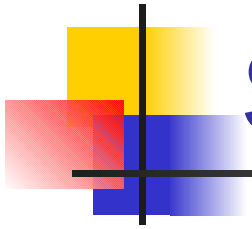


- Formulate a theory of mind/ brain
- Express the theory in a computer program
- Two Approaches
  - Cognitive Science and Psychology (testing/ predicting responses of human subjects)
  - Cognitive Neuroscience (observing neurological data)



Systems Thinking Rationally

- "Rational" -> **ideal** intelligence
  (contrast with **human** intelligence)
- Rational thinking governed by precise "laws of thought"
  - syllogisms
  - notation and logic
- Systems (in theory) can solve problems using such laws



Systems Acting Rationally

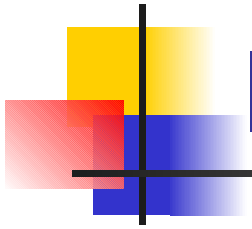
Building systems that carry out actions to achieve the **best outcome**

Rational **behavior**

May or may not involve rational thinking

- i.e., consider reflex actions

This is the definition we will adopt

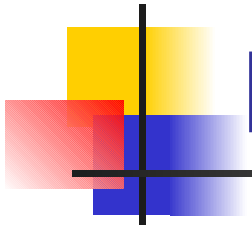


Intelligent Agents

Agent: anything that perceives and acts on its environment

AI: study of rational agents

A rational agent carries out an action with the **best outcome** after considering past and current percepts



Foundations of AI

Philosophy: logic, mind, knowledge

Mathematics: proof, computability, probability

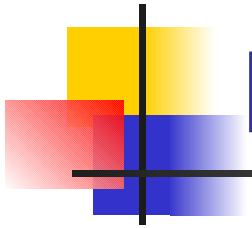
Economics: maximizing payoffs

Neuroscience: brain and neurons

Psychology: thought, perception, action

Control Theory: stable feedback systems

Linguistics: knowledge representation, syntax



Brief History of AI

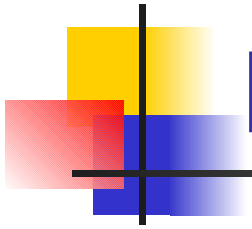
1943: McCulloch & Pitts: Boolean circuit model of brain

1950: Turing's “Computing Machinery and Intelligence”

1952—69: Look, Ma, no hands!

1950s: Early AI programs, including Samuel's checkers program, Newell & Simon's Logic Theorist, Gelernter's Geometry Engine

1956: Dartmouth meeting: “Artificial Intelligence” adopted



Brief History of AI

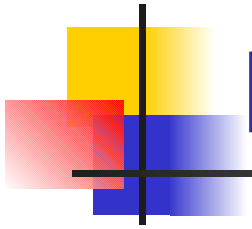
1965: Robinson's complete algorithm for logical reasoning

1966—74: AI discovers computational complexity; Neural network research almost disappears

1969—79: Early development of knowledge- based systems

1980—88: Expert systems industry booms

1988—93: Expert systems industry busts: `”AI Winter”



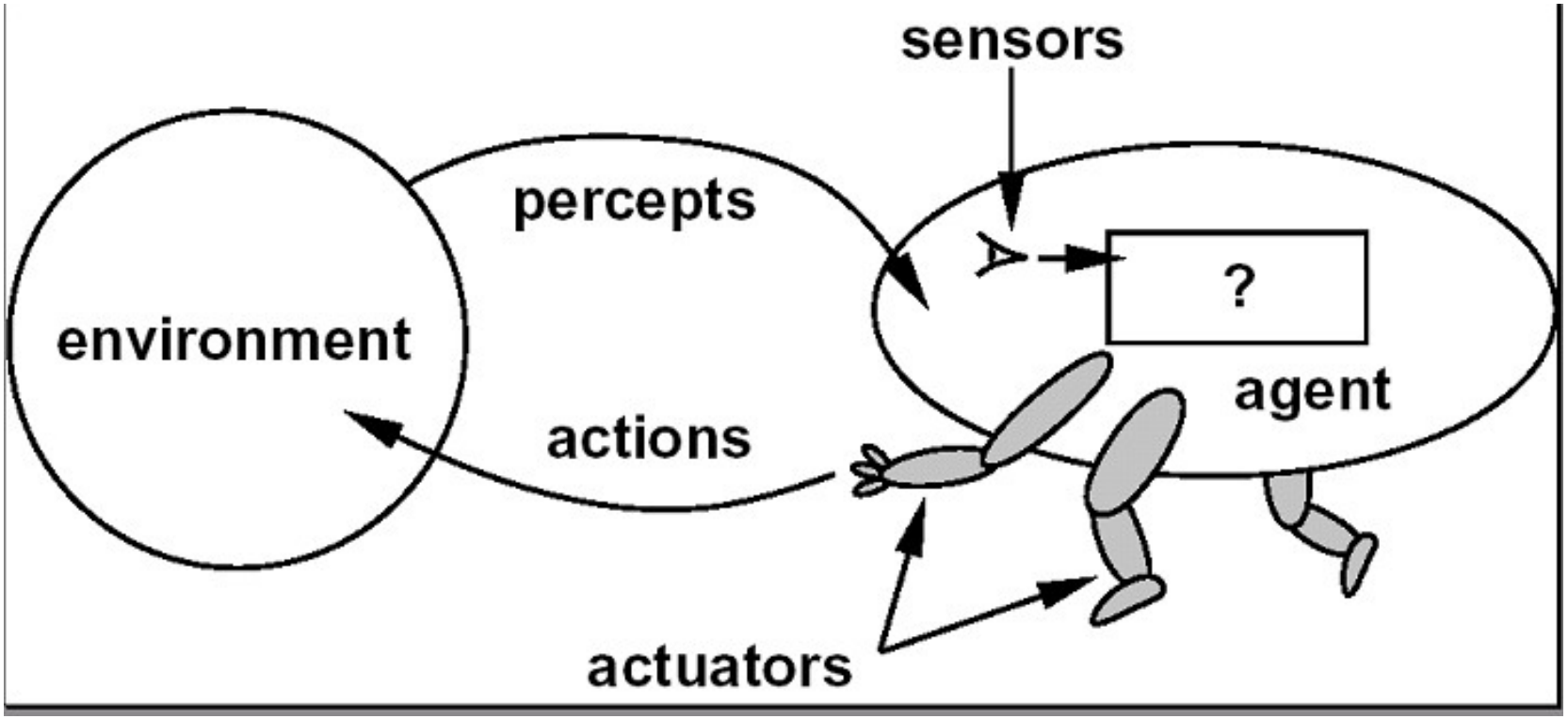
Brief History of AI

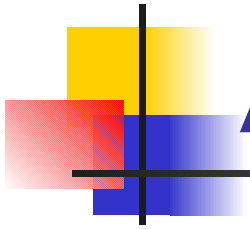
1985—95: Neural networks return to popularity

1988— Resurgence of probability; general increase in technical depth, "Nouvelle AI": ALife, GAs, soft computing

1995— Agents...

Back to Agents





Agent Function

$a = F(p)$

where p is the current percept, a is the action carried out, and F is the agent function

F maps percepts to actions

$F: P \rightarrow A$

where P is the set of all percepts, and A is the set of all actions

In general, an action may depend on all percepts observed so far, not just the current percept, so…

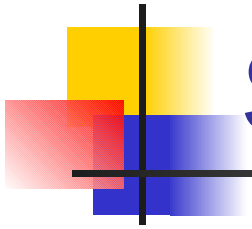
Agent Function Refined

$a_k = F(p_0\, p_1\, p_2 \ldots p_k)$

where $p_0\, p_1\, p_2 \ldots p_k$ is the sequence of percepts observed to date, a_k is the resulting action carried out

F now maps *percept sequences* to actions

$F: P^* \rightarrow A$



Structure of Agents

Agent = architecture + program

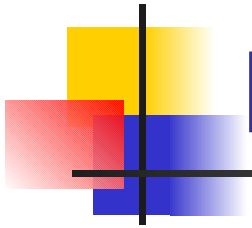
architecture

- device with sensors and actuators
- e.g., A robotic car, a camera, a PC, ...

program

- implements the agent function on the architecture

Specifying the Task Environment



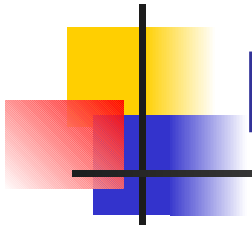
PEAS

Performance Measure: captures agent's aspiration

Environment: context, restrictions

Actuators: indicates what the agent can carry out

Sensors: indicates what the agent can perceive



Properties of Environments

Fully versus partially observable

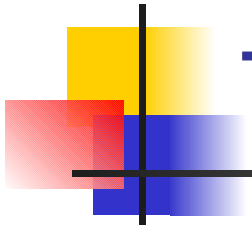
Deterministic versus stochastic

Episodic versus sequential

Static versus dynamic

Discrete versus continuous

Single agent versus multiagent



Types of Agents

Reflex Agent

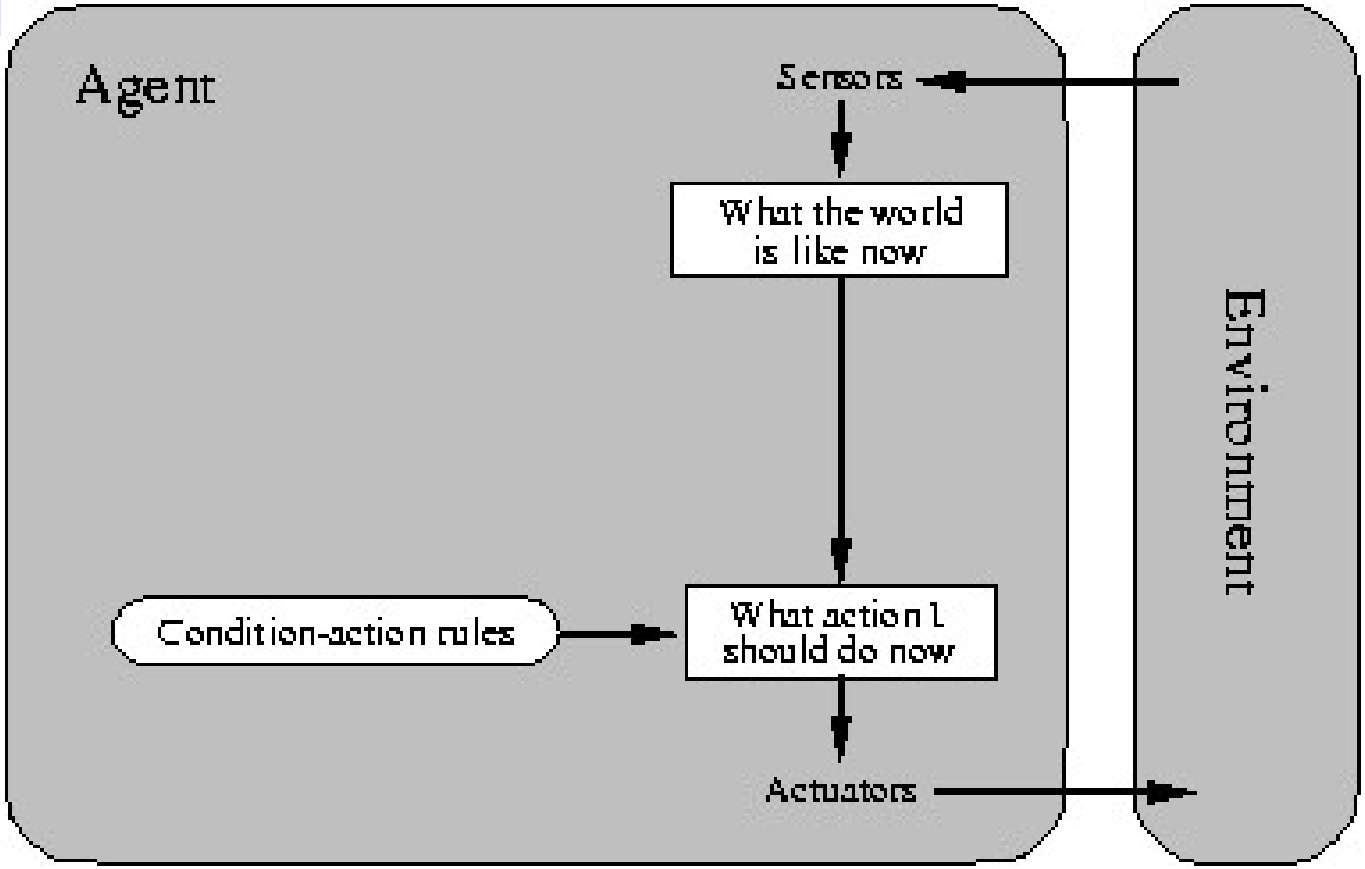
Reflex Agent with State

Goal-based Agent

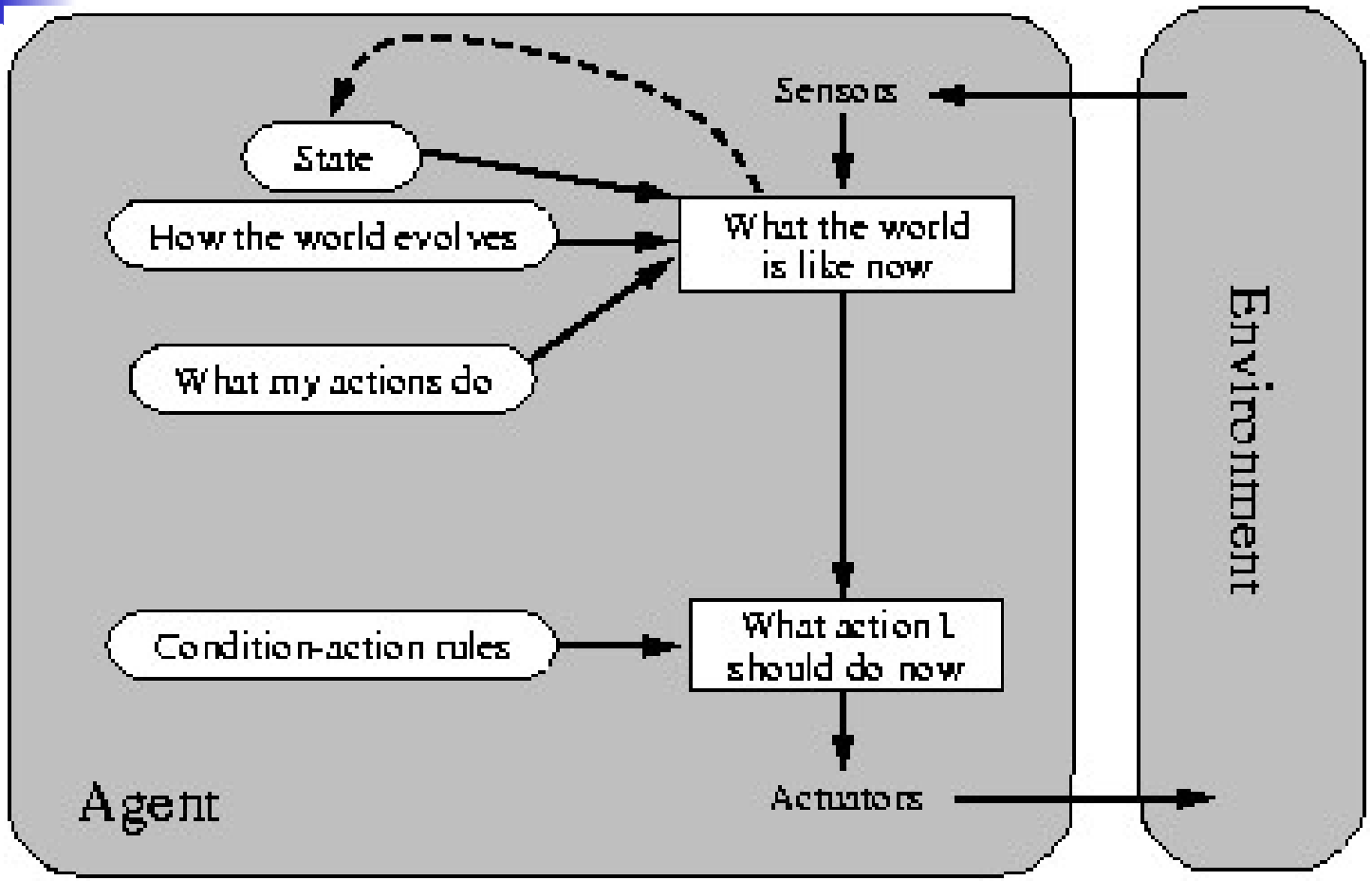
Utility-Based Agent

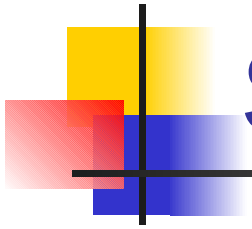
Learning Agent

Reflex Agent



Reflex Agent with State





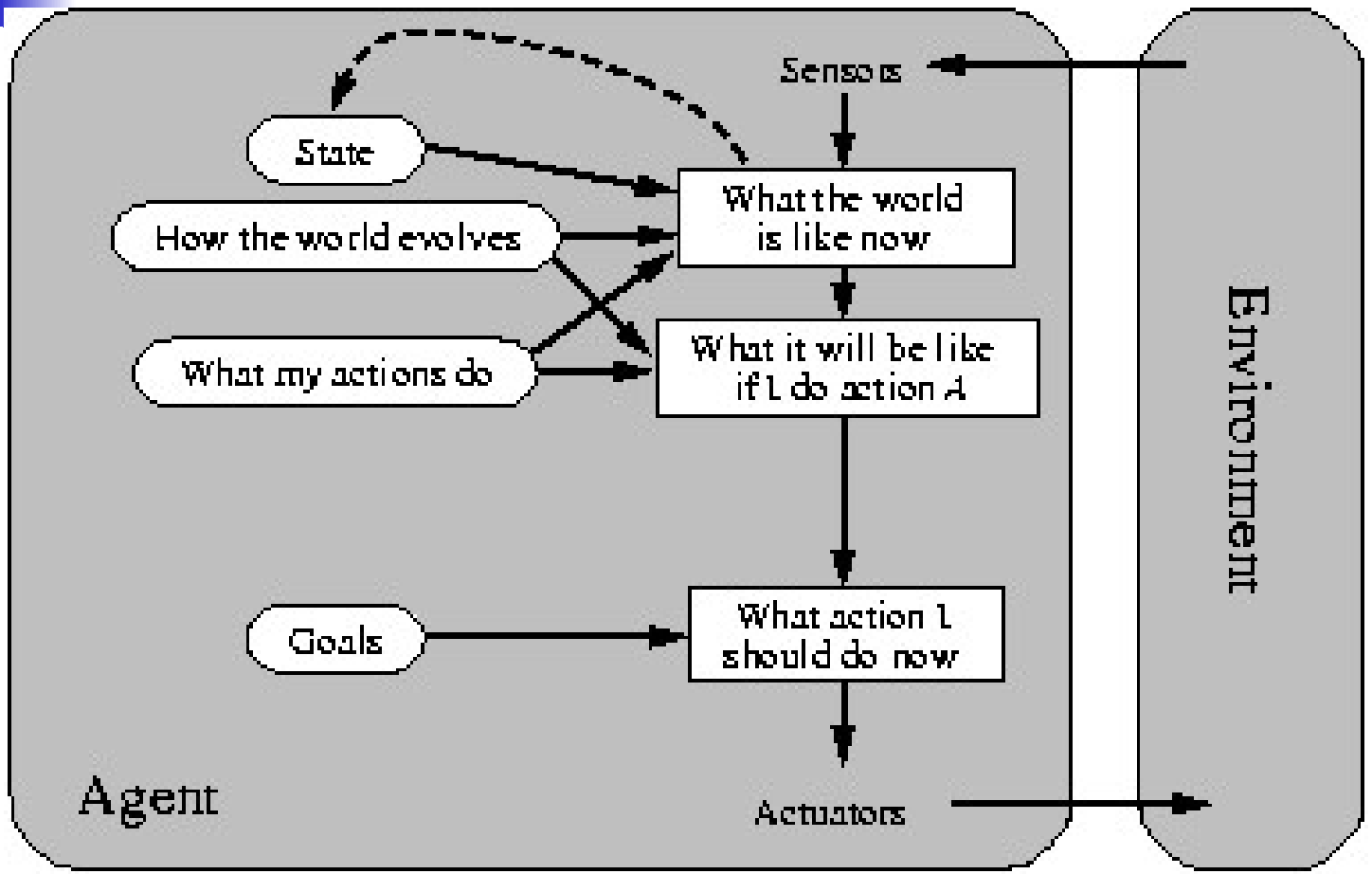
State Management

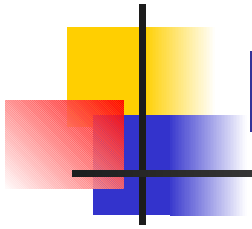
Reflex agent with state

- Incorporates a **model** of the world
- Current state of its world depends on percept history
- Rule to be applied next depends on resulting state

state' ← next-state(state, percept)
action ← select-action(state', rules)

Goal- based Agent

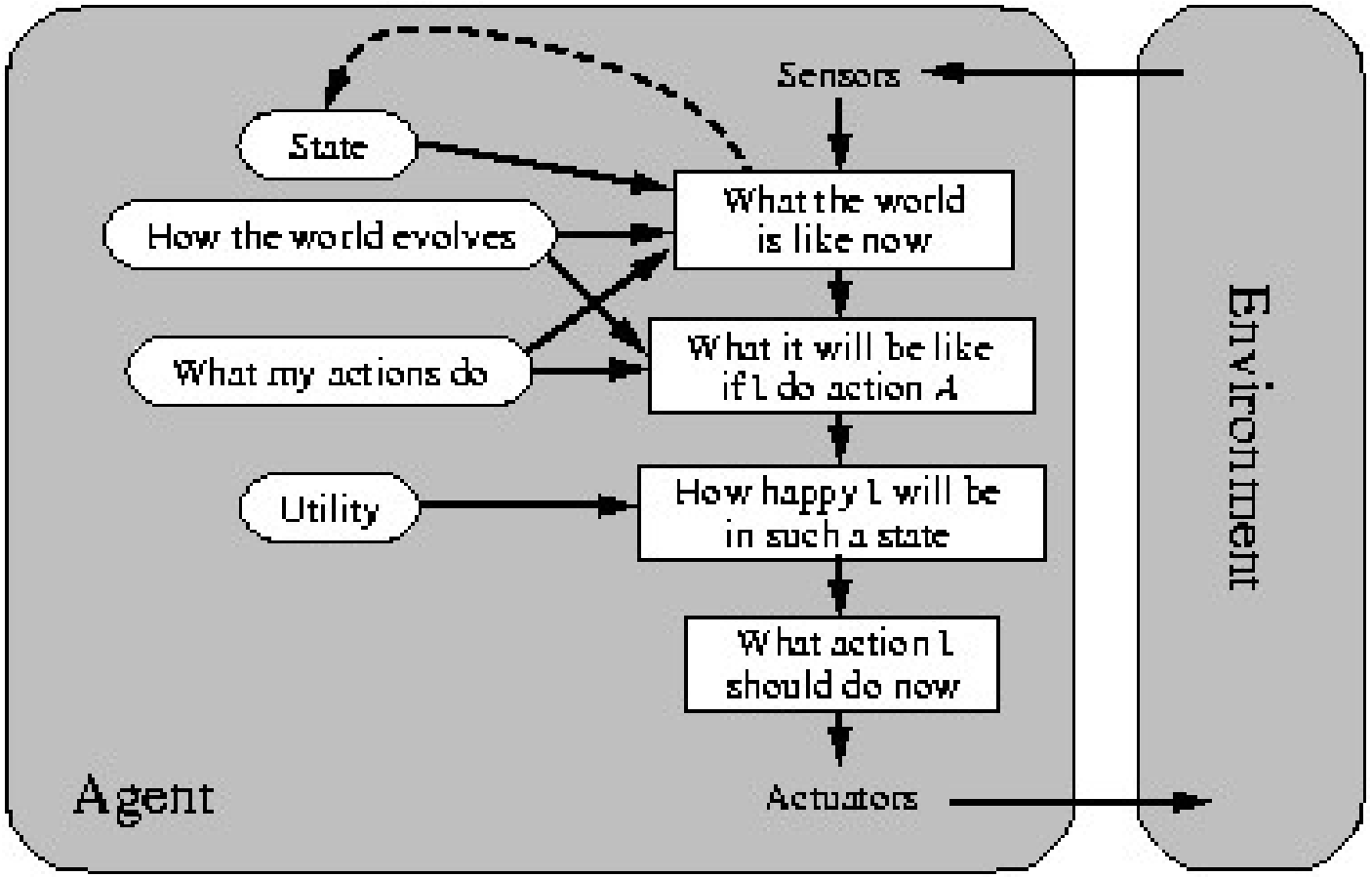


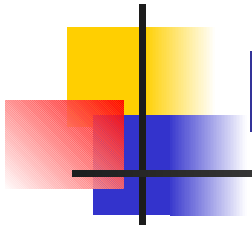


Incorporating Goals

- Rules and “foresight”
  - Essentially, the agent’s rule set is determined by its goals
  - Requires knowledge of future consequences given possible actions
- Can also be viewed as an agent with more complex state management
  - Goals provide for a more sophisticated next-state function

Utility- based Agent

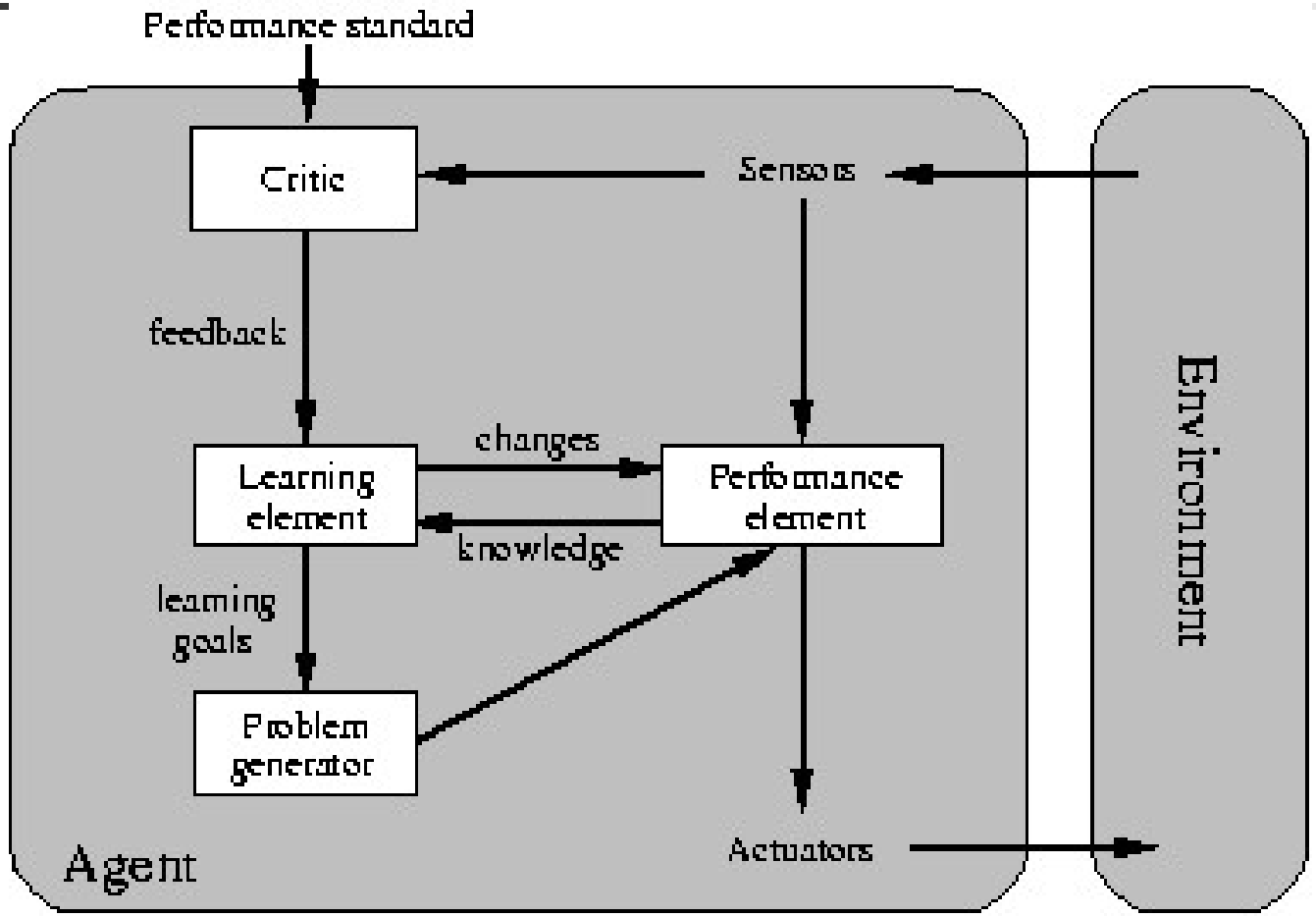


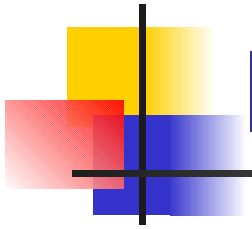


Incorporating Performance

- May have multiple action sequences that arrive at a goal
- Choose action that provides the best level of “happiness” for the agent
- Utility function maps states to a measure
  - May include tradeoffs
  - May incorporate likelihood measures

Learning Agent





Incorporating Learning

- Can be applied to any of the previous agent types
  - Agent <-> Performance Element
- Learning Element
  - Causes improvements on agent/ **performance element**
  - Uses feedback from **critic**
  - Provides goals to **problem generator**

Next: Problem Solving Agents (Chap 3- 6)

```
function SIMPLE-PROBLEM-SOLVING-AGENT(percept) returns an action
    static: seq, an action sequence, initially empty
            state, some description of the current world state
            goal, a goal, initially null
            problem, a problem formulation

    state ← UPDATE-STATE(state, percept)
    if seq is empty then do
        goal ← FORMULATE-GOAL(state)
        problem ← FORMULATE-PROBLEM(state, goal)
        seq ← SEARCH(problem)
    action ← FIRST(seq)
    seq ← REST(seq)
    return action
```