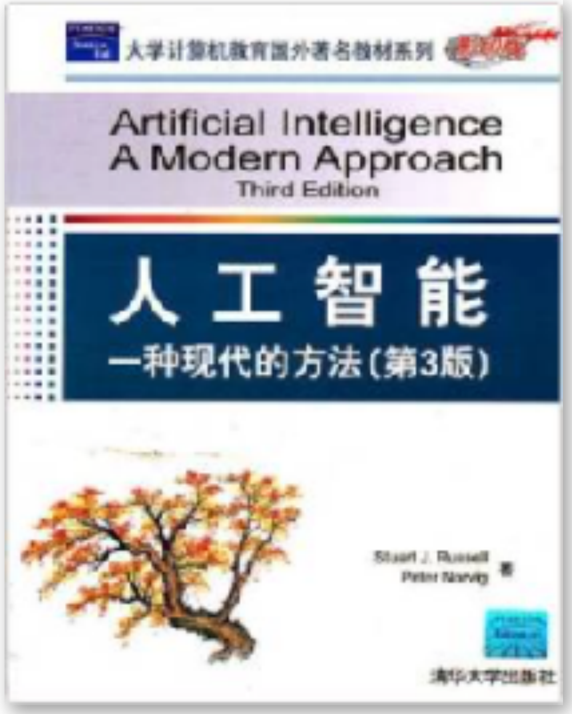
关于本课

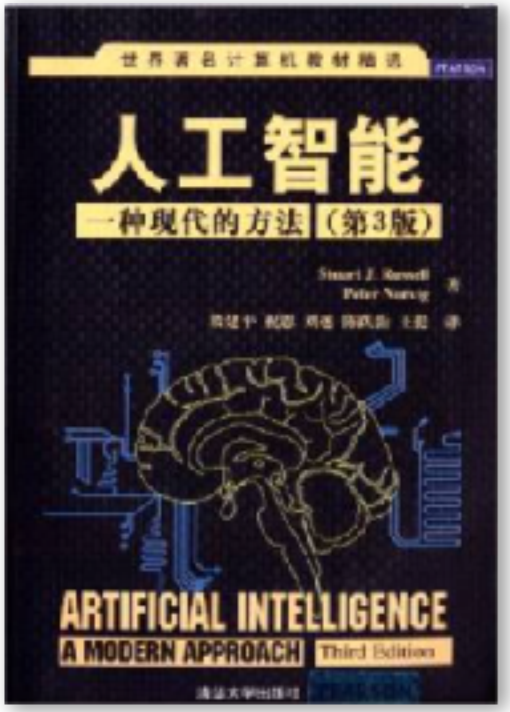
课程讨论QQ群： 196685563

教材

课程名称：人工智能

教材：AIMA





http://aima.cs.berkeley.edu/

课程主页

时间： 周四 14:00-16:00 仙II-304

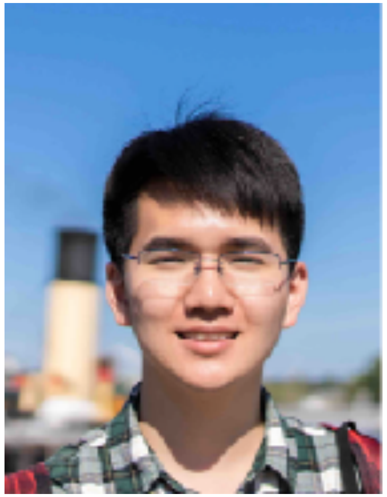
课程主页：http://lamda.nju.edu.cn/IntroAI19/

http://lamda.nju.edu.cn/IntroAI19/course_page.html

助教

秦熔均

胡圣佑



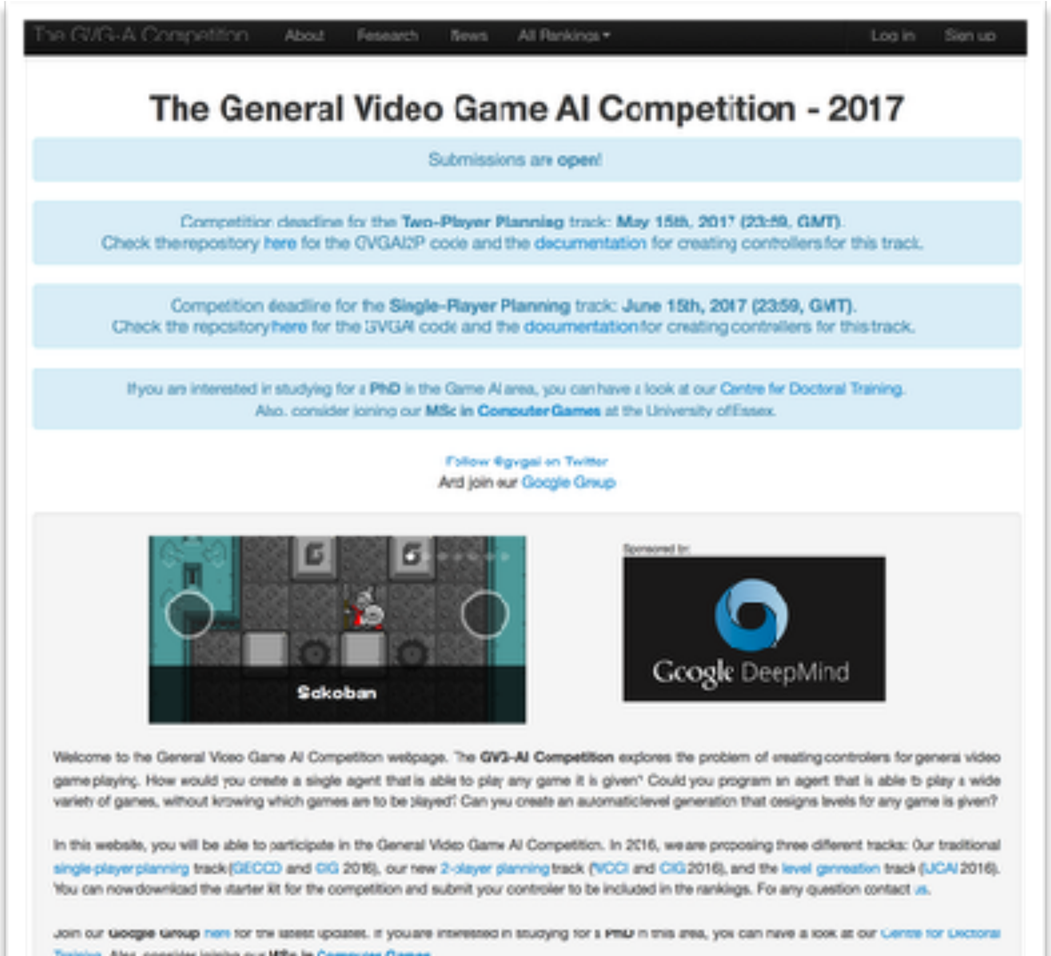
刘驭壬

作业

本次课程有五次作业--让计算机自己玩游戏

将基于GVGAI框架，请开始熟悉该框架：
http://www.gvgai.net

作业5次
每次占 16%，共80%
期末考试：20%



The General Video Game AI Competition - 2017

Submissions are **open**!

Competition deadline for the **Two-Player Planning** track: **May 15th, 2017 (23:59, GMT)**.
Check the repository **here** for the GVGAI2P code and the **documentation** for creating controllers for this track.

Competition deadline for the **Single-Player Planning** track: **June 15th, 2017 (23:59, GMT)**.
Check the repository **here** for the GVGAI code and the **documentation** for creating controllers for this track.

If you are interested in studying for a **PhD** in the Game AI area, you can have a look at our Centre for Doctoral Training.
Also, consider joining our **MSc in Computer Games** at the University of Essex.

Follow @gvgai on Twitter
And join our Google Group

Welcome to the General Video Game AI Competition webpage. The **GVG-AI Competition** explores the problem of creating controllers for general video game playing. How would you create a single agent that is able to play any game it is given? Could you program an agent that is able to play a wide variety of games, without knowing which games are to be played? Can you create an automatic level generation that designs levels for any game is given?

In this website, you will be able to participate in the General Video Game AI Competition. In 2016, we are proposing three different tracks: Our traditional single-player planning track (GECCO and CIG 2016), our new 2-player planning track (WCCI and CIG 2016), and the level generation track (IJCAI 2016). You can now download the starter kit for the competition and submit your controller to be included in the rankings. For any question contact us.

Join our **Google Group** here for the latest updates. If you are interested in studying for a **PhD** in this area, you can have a look at our Centre for Doctoral Training. Also, consider joining our **MSc in Computer Games**.

Lecture 1: Introduction

What is artificial intelligence?

1956 Dartmouth meeting: "Artificial Intelligence"

John McCarthy:

> " It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable."

1927-2011

Marvin Minsky:

> " to make computers be capable of doing things that when done by a human, would be thought to require intelligence "

1927-2016

we will discuss the concept and the history of AI in the last class

What we call AI in movies

2001: A Space Odyssey
1968

The Matrix
1999

A.I. Artificial Intelligence
2001

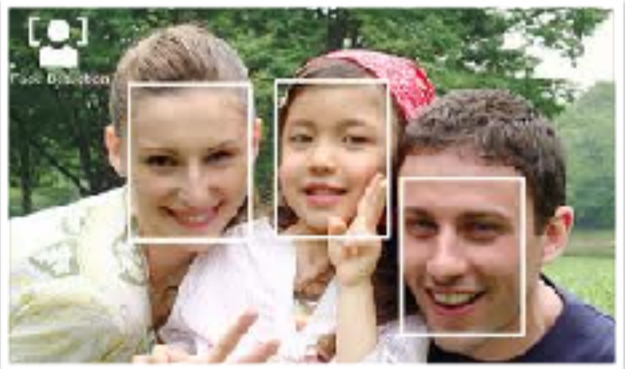
Wall-E
2008

I, Robot
2004

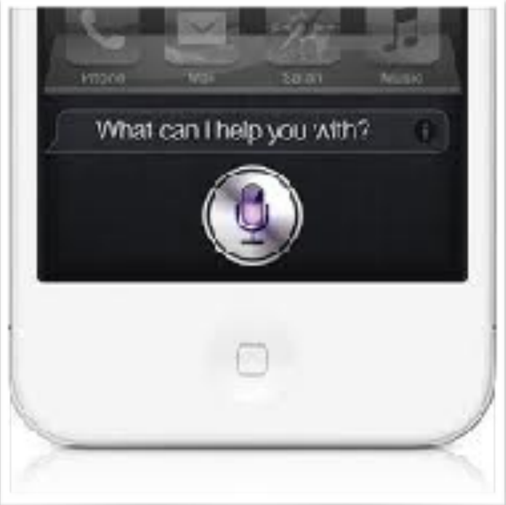
Interstellar
2014

The Terminator
1984

What AI we do have



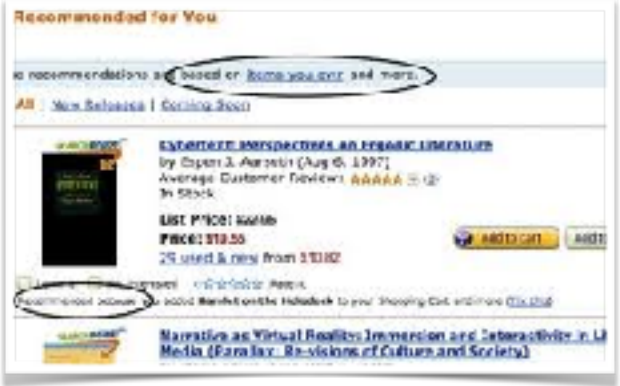
人脸检测、识别



What can I help you with?

S.I.R.I.

自动驾驶



Recommended for You

...recommendations are based on items you own and more.

All | New Releases | Coming Soon

Cybertext: Perspectives on Ergodic Literature
by Espen J. Aarseth (Aug 6, 1997)
Average Customer Review: ★★★★★
In Stock

LIST PRICE: [illegible]
PRICE: $19.55
25 used & new from $10.02

Add to cart

Recommended because you added Hamlet on the Holodeck to your Shopping Cart and more (Fix this)

Narrative as Virtual Reality: Immersion and Interactivity in Li...
Media (Parallax: Re-visions of Culture and Society)

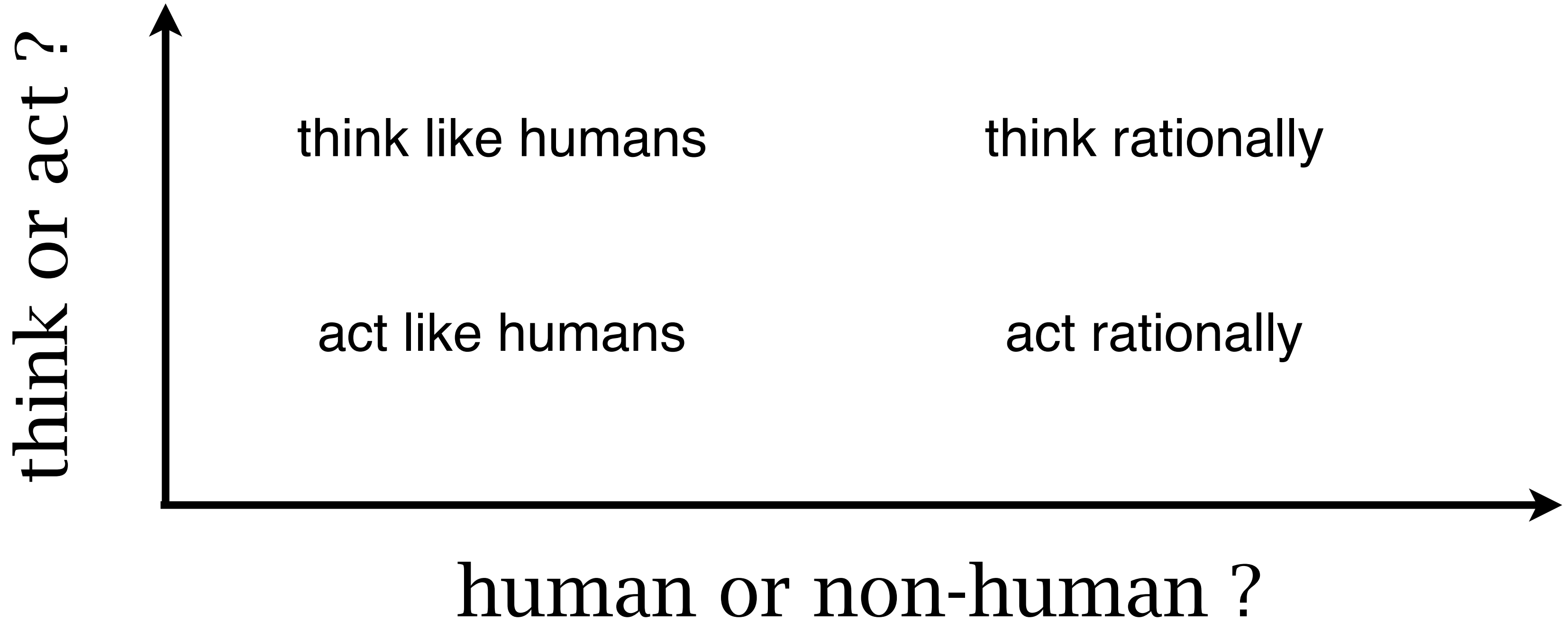
推荐系统

BigDog

下棋

What is AI?

AI is a system that



think or act ?

think like humans

think rationally

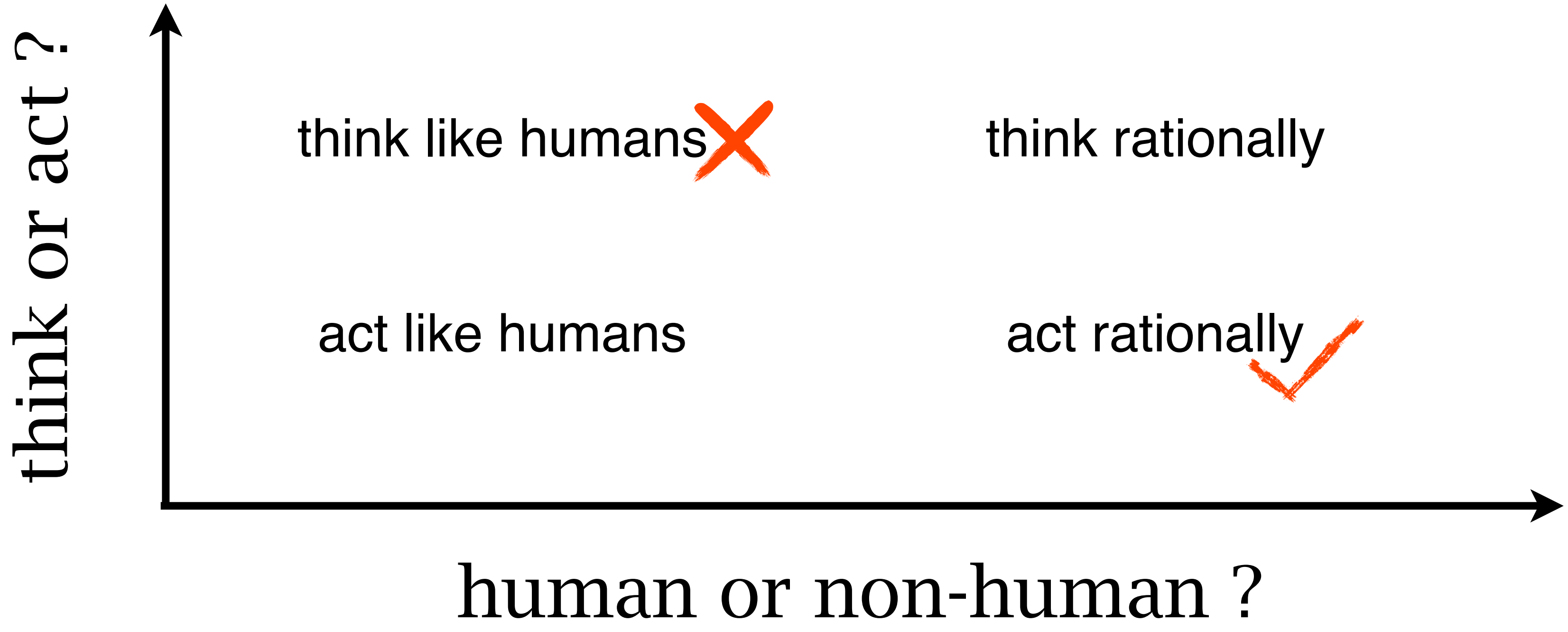
act like humans

act rationally

human or non-human ?

What is AI?

AI is a system that



think or act ?

think like humans ✗

think rationally

act like humans

act rationally ✓

human or non-human ?

Current top AI systems

AlphaGo

2016年3月，AlphaGo 战胜韩国职业选手李世乭（九段）

2017年1月初，快棋版本 Master 取得60:0战绩

Current top AI systems

DeepStack & Libratus

2017年1月左右，在一对一无限注德州扑克上大幅赢过职业选手

What we will learn

Search 搜索与规划

Knowledge 知识表达与处理

Uncertainty 不确定建模

Learning 机器学习

What we will do

Search 搜索与规划

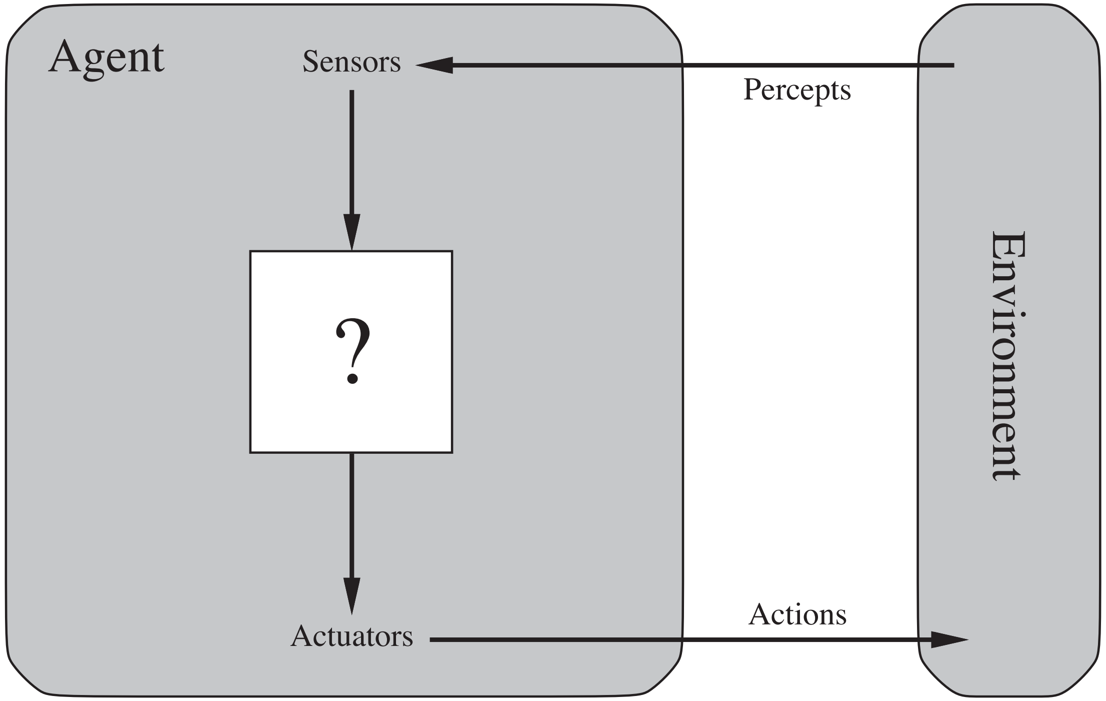
Knowledge 知识表达与处理

Uncertainty 不确定建模

Learning 机器学习

⇒ General Game Player

Agent



Agents include humans, robots, softbots, thermostats, etc.

The agent function maps from percept histories to actions:

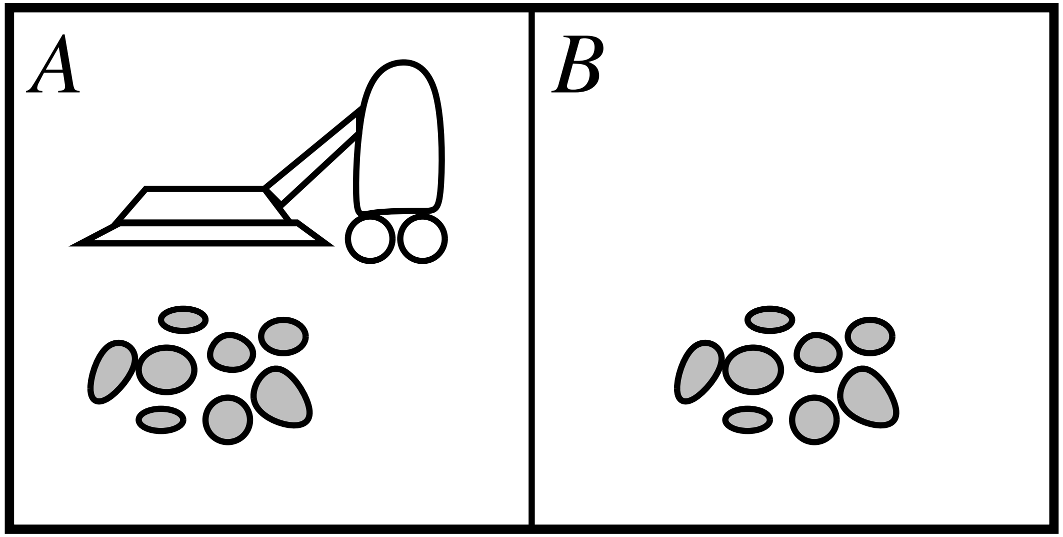
$$f : \mathcal{P}^* \to \mathcal{A}$$

The agent program runs on the physical architecture to produce f

Example: Vacuum-cleaner world

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

Actions: $Left$, $Right$, $Suck$, $NoOp$



A vacuum-cleaner agent

Percept sequence	Action
$[A, Clean]$	$Right$
$[A, Dirty]$	$Suck$
$[B, Clean]$	$Left$
$[B, Dirty]$	$Suck$
$[A, Clean]$, $[A, Clean]$	$Right$
$[A, Clean]$, $[A, Dirty]$	$Suck$
⋮	⋮

```
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
```

What is the **right** function?
Can it be implemented in a small agent program?

P. E. A. S.

To design an agent, we need to specify **four-dimensions**:

Performance measure?

Environment?

Actuators?

Sensors?

Examples of PEAS

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display of scene categorization	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, suggestions, corrections	Keyboard entry

Environment types

In six-dimensions:

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
Interactive English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

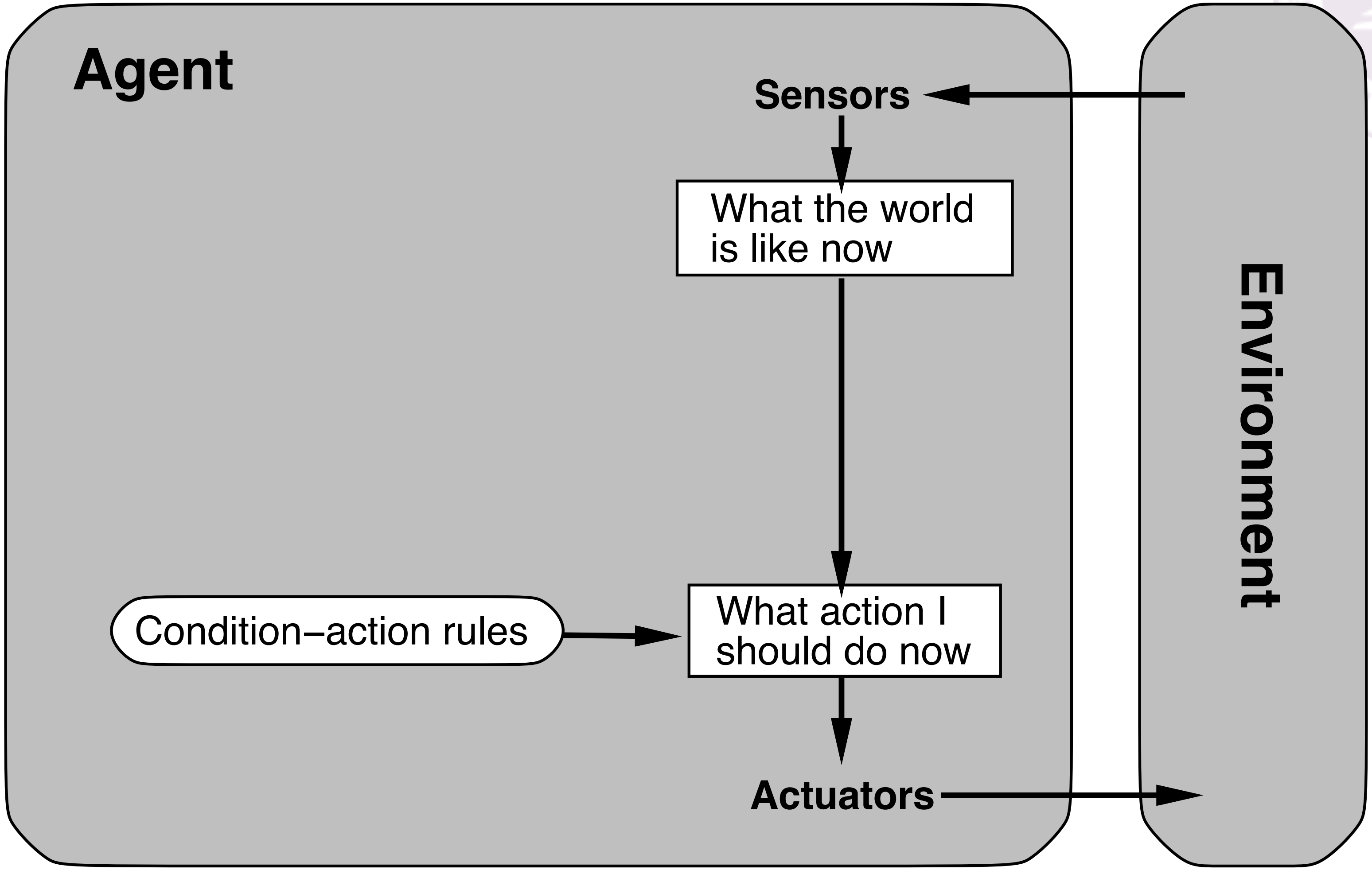
Agent types

Four basic types in order of increasing generality:

- simple reflex agents
- reflex agents with state
- goal-based agents
- utility-based agents

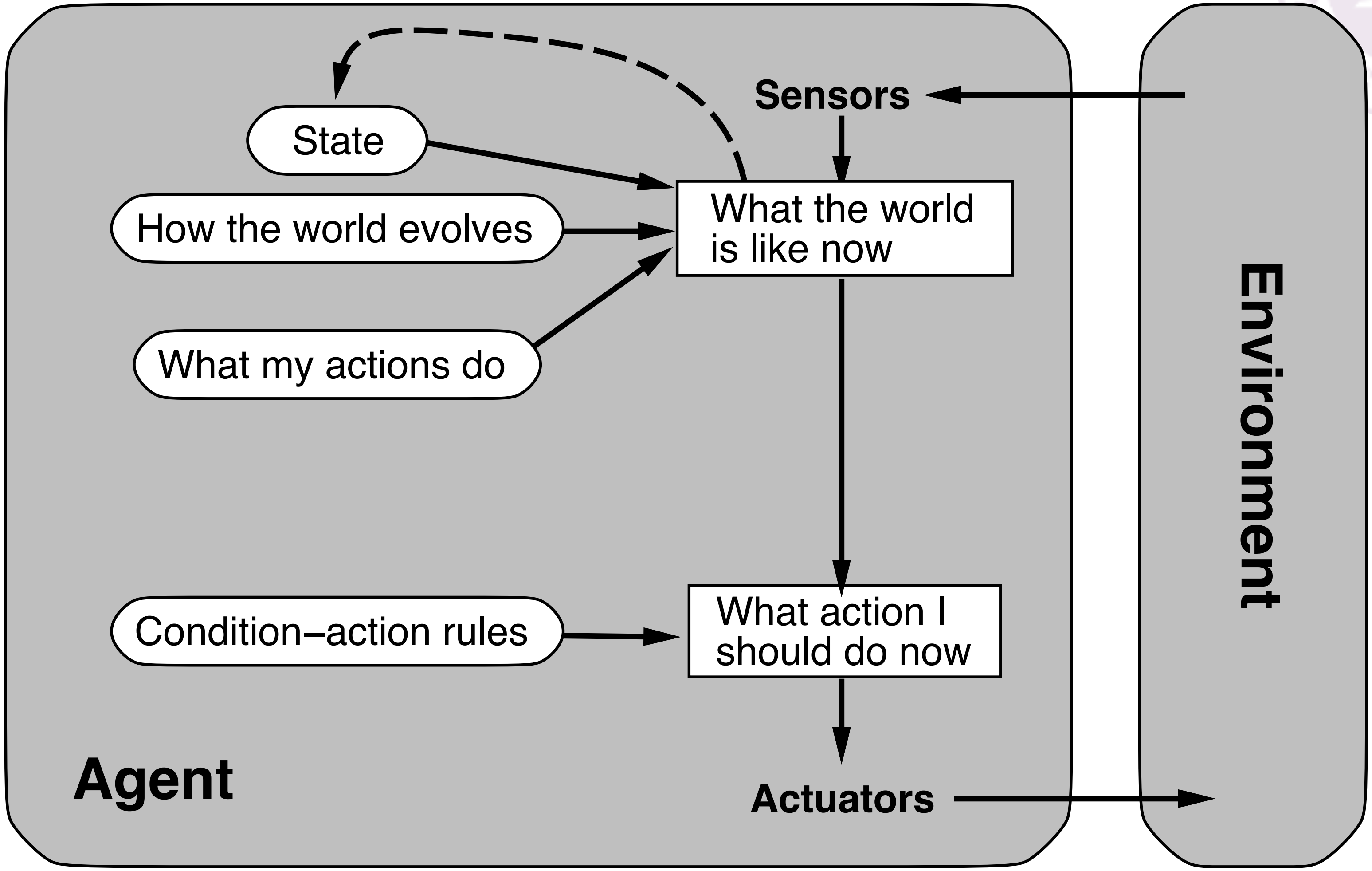
All these can be turned into learning agents

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 $Left$
```

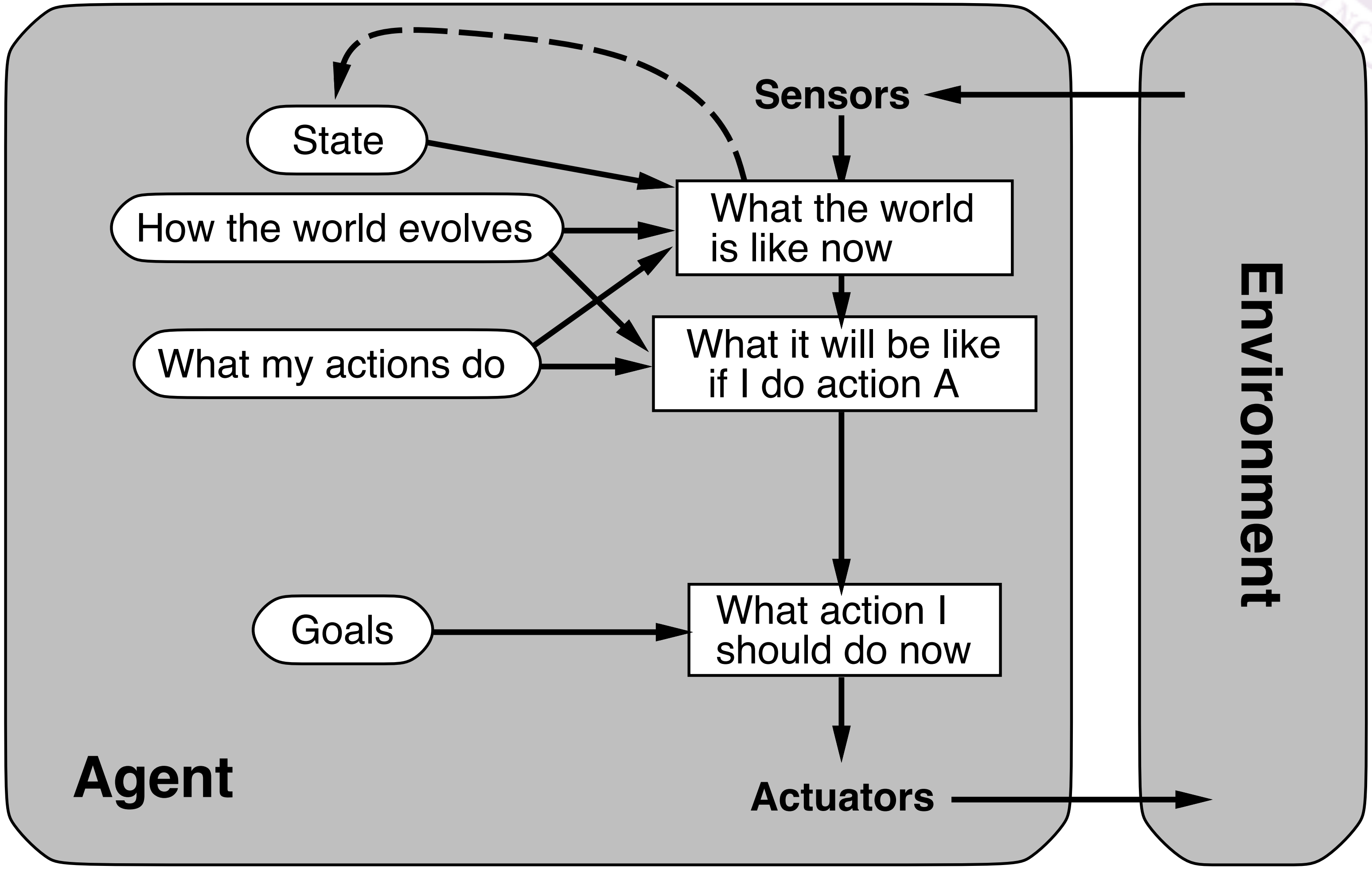
Reflex agents with state



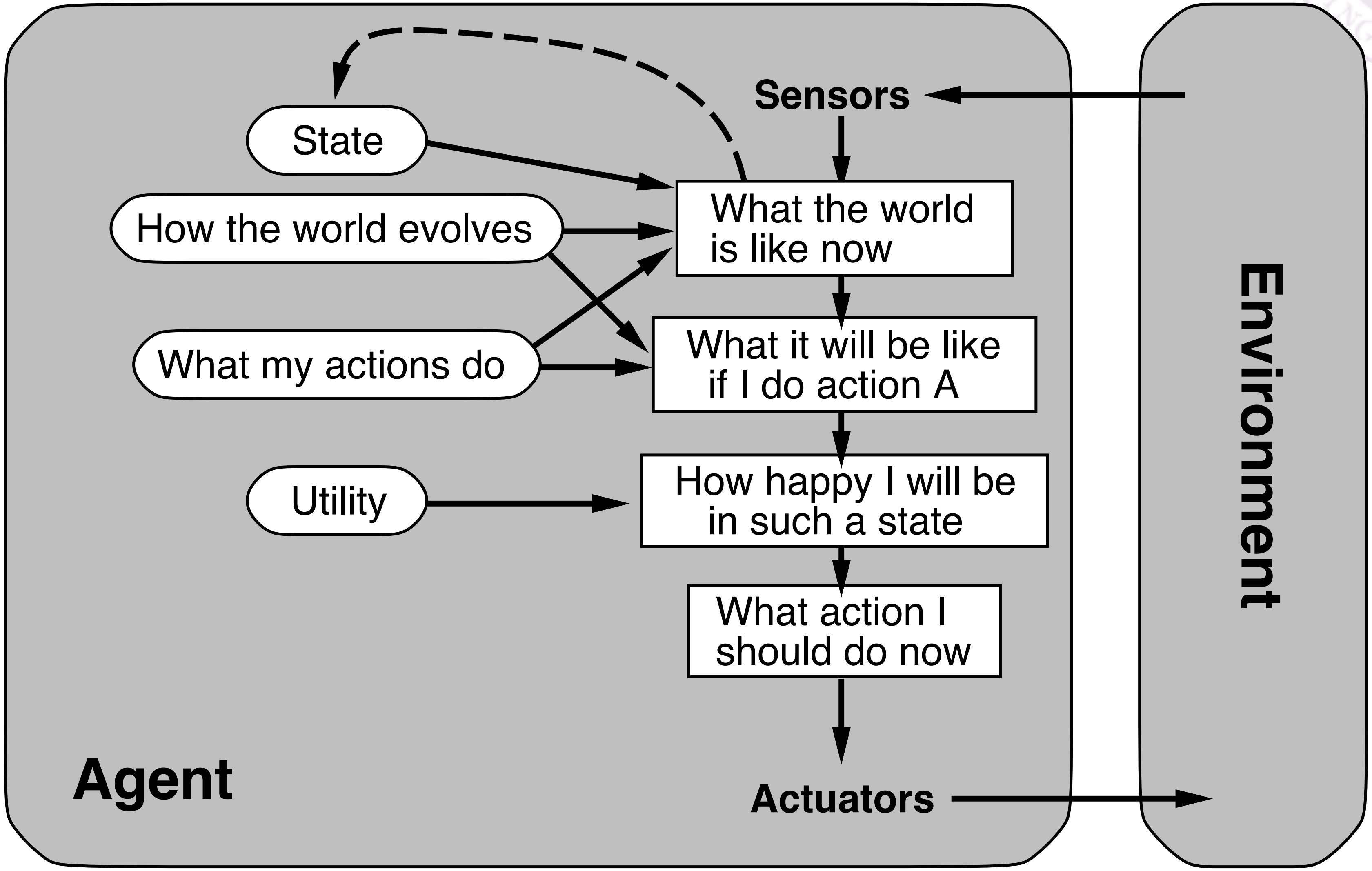
```
function REFLEX-VACUUM-AGENT([$location$, $status$]) returns an action
static: $last\_A$, $last\_B$, numbers, initially $\infty$

    if $status = Dirty$ then . . .
```

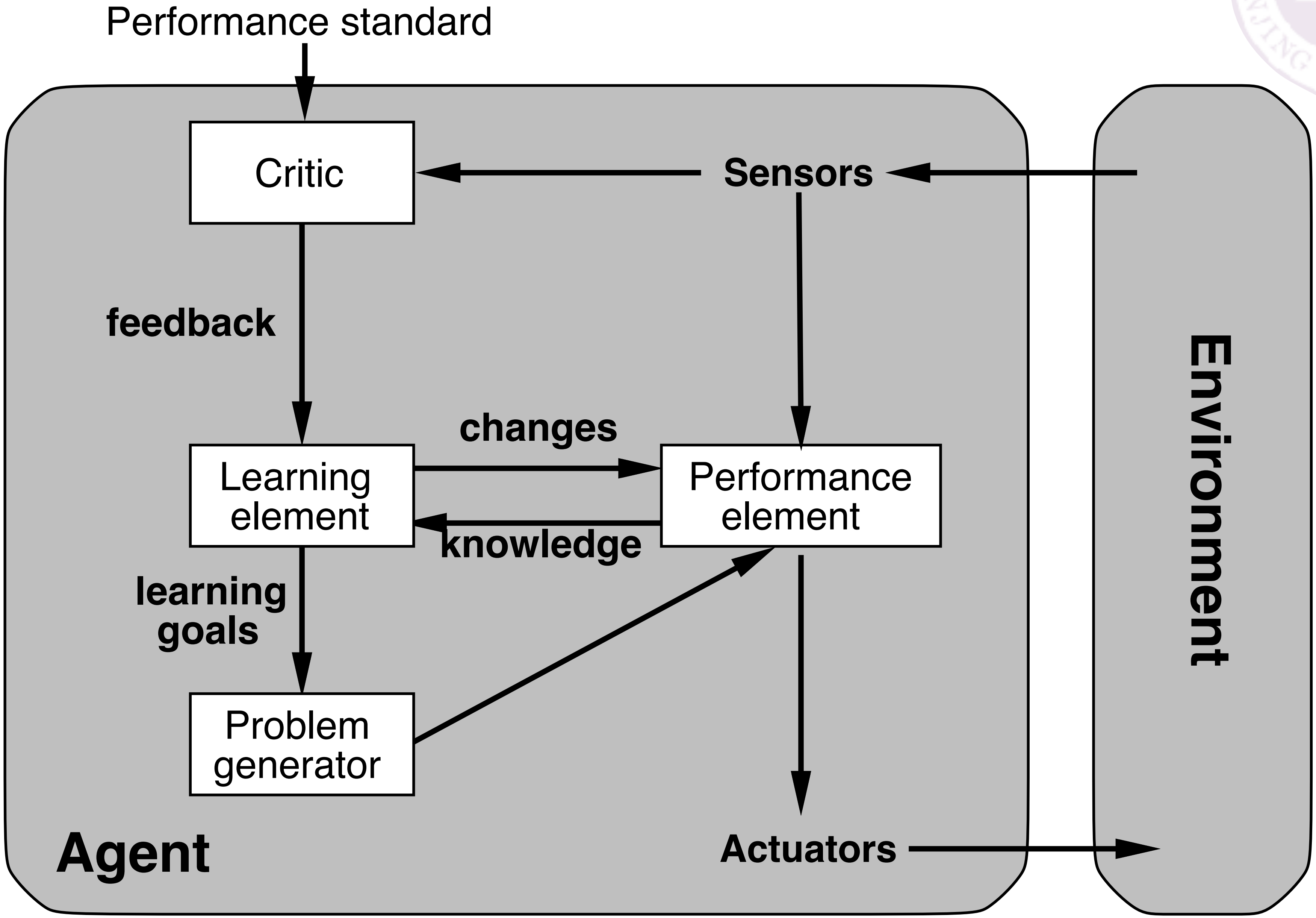
Goal-based agents


State
How the world evolves
What my actions do
Goals
Sensors
What the world is like now
What it will be like if I do action A
What action I should do now
Actuators
Agent
Environment

Utility-based agents



Learning agents


Performance standard
Critic
Sensors
feedback
changes
Learning element
Performance element
knowledge
learning goals
Problem generator
Environment
Agent
Actuators