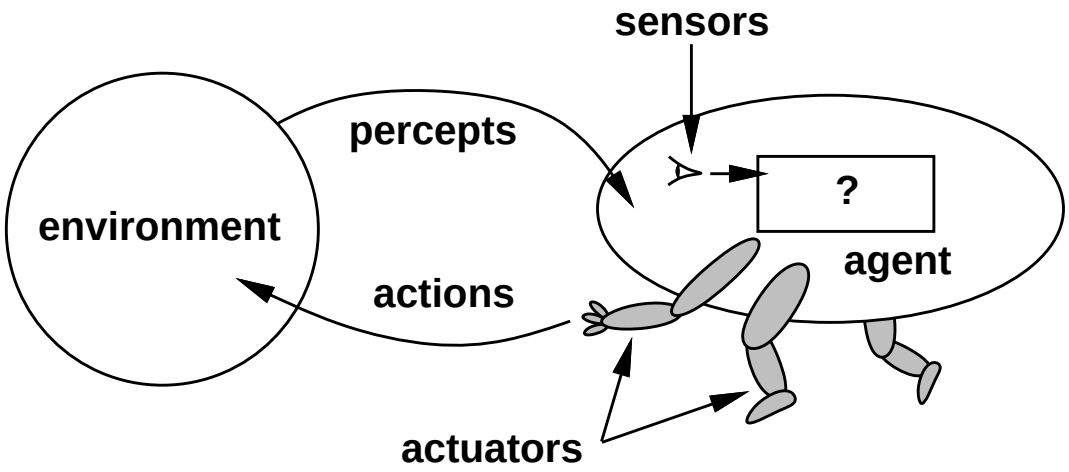
Intelligent Agents

Chapter 2

Outline

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

Agents and environments



percepts via sensors

acts via actuators

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

Percept refers to the agent perceptual input at any given instant

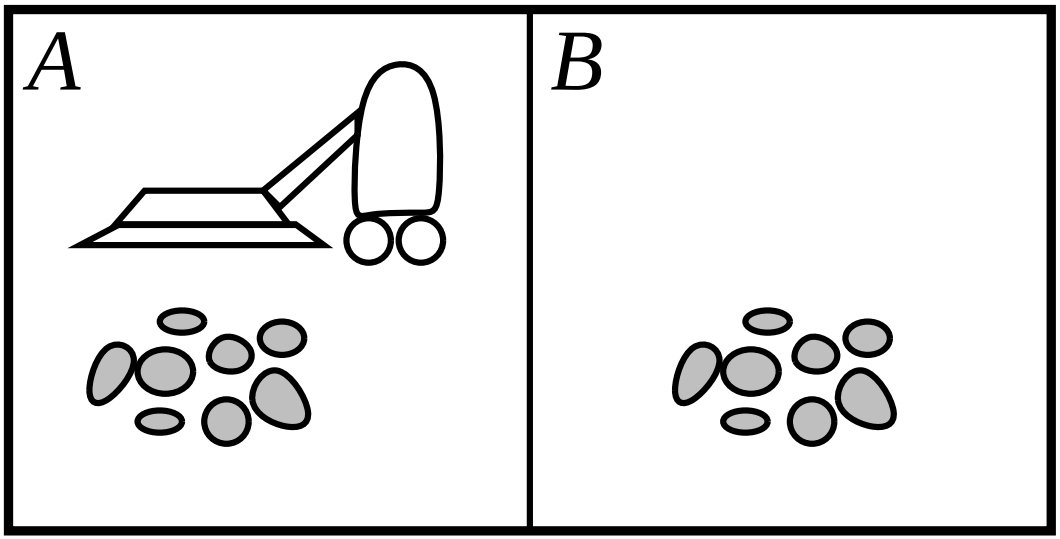
The agent function maps from percept histories to actions:

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

[sequence of elements]

The agent program implements f on the physical architecture.

Vacuum-cleaner world



Percepts: current location and its content, e.g., $(A, Dirty)$

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

A← →B clean stop

A vacuum-cleaner agent

Percept sequence	Action
$(A, Clean)$	$Right$
$(A, Dirty)$	$Suck$
$(B, Clean)$	$Left$
$(B, Dirty)$	$Suck$
$(A, Clean), (A, Clean)$	$Right$
$(A, Clean), (A, Dirty)$	$Suck$
⋮ only look at the current state	⋮

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
```

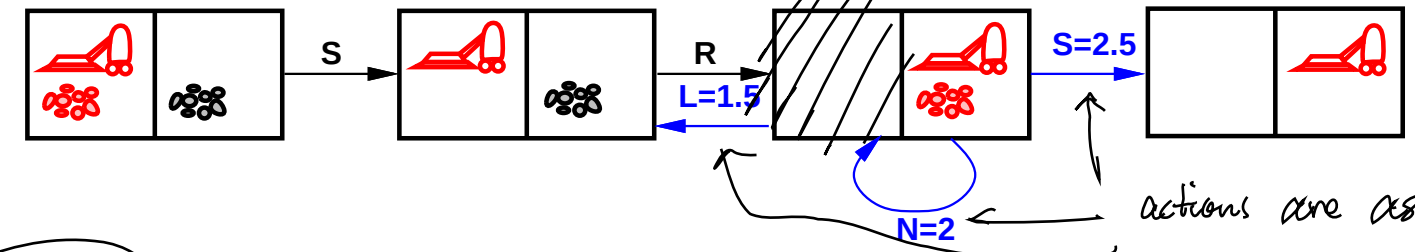
What is the **right** function f?

Can it be implemented in a small agent program?

Rationality

The performance measure evaluates the environment sequence

- one point per room cleaned up within T time steps?
- one point per clean room per time step, minus half a point per action?
- penalize for $> k$ dirty rooms?



A rational agent chooses whichever action maximizes the expected value of the performance measure given the percept sequence to date

Rational $\neq$ omniscient

- percepts may not supply all relevant information

Rational $\neq$ clairvoyant

- action outcomes may not be as expected

Hence, rational $\neq$ successful

PEAS

To design a rational agent, we must specify the task environment

Consider, e.g., the task of designing a driverless taxi:

Performance measure?? < what to persuit / optimize on

Environment?? < what to consider

Actuators??

Sensors??

PEAS

To design a rational agent, we must specify the task environment

Consider, e.g., the task of designing a driverless taxi:

Performance measure?? safety, destination, profits, legality, comfort, . . .

Environment?? streets/freeways, traffic, pedestrians, weather, . . .

Actuators?? steering, accelerator, brake, horn, blinkers, . . .

Sensors?? GPS, video, accelerometers, gauges, engine sensors, . . .

Internet shopping agent

Performance measure??

Environment??

Actuators??

Sensors??

Internet shopping agent

Performance measure?? price, quality, appropriateness, efficiency

Environment?? user, WWW sites, vendors, shippers

Actuators?? display to user, follow URL, fill in form

Sensors?? HTML pages (text, graphics, scripts), user input

Properties of Task Environments

Fully vs partially observable: do the agent sensors give access to all relevant information about the environment state?

Deterministic vs stochastic: is the next state completely determined by the current state and executed action? [or there are other influential factors]

Known vs unknown?: does the agent know the environment's laws of physics?

Episodic vs sequential: is the next decision independent of the previous ones?

Static vs dynamic: can the environment change whilst the agent is deliberating? Semi-dynamic: only the performance score changes.

Discrete vs continuous: can time, states, actions, percepts be represented in a discrete way?

Single vs multi-agent: is a single agent making decisions, or do multiple agents need to compete or cooperate to maximise inter-dependent performance measures?

Environment types

	Puzzle	Poker	Part picking robot	Taxi
Observable?? Deterministic?? known?? Episodic?? Static?? Discrete?? Single-agent??				

Environment types

	Puzzle	Poker	Part picking robot	Taxi
Observable??	Yes	No (can't see opp.)	Mostly	No
Deterministic??	Yes	No	Partly	No
known??	Yes	Yes	Yes	Partly
Episodic??	No	No	Yes	No
Static??	Yes	Yes	No	No
Discrete??	Yes	Yes	No	No
Single-agent??	Yes	No	Yes	No

The environment type largely determines the agent design

The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

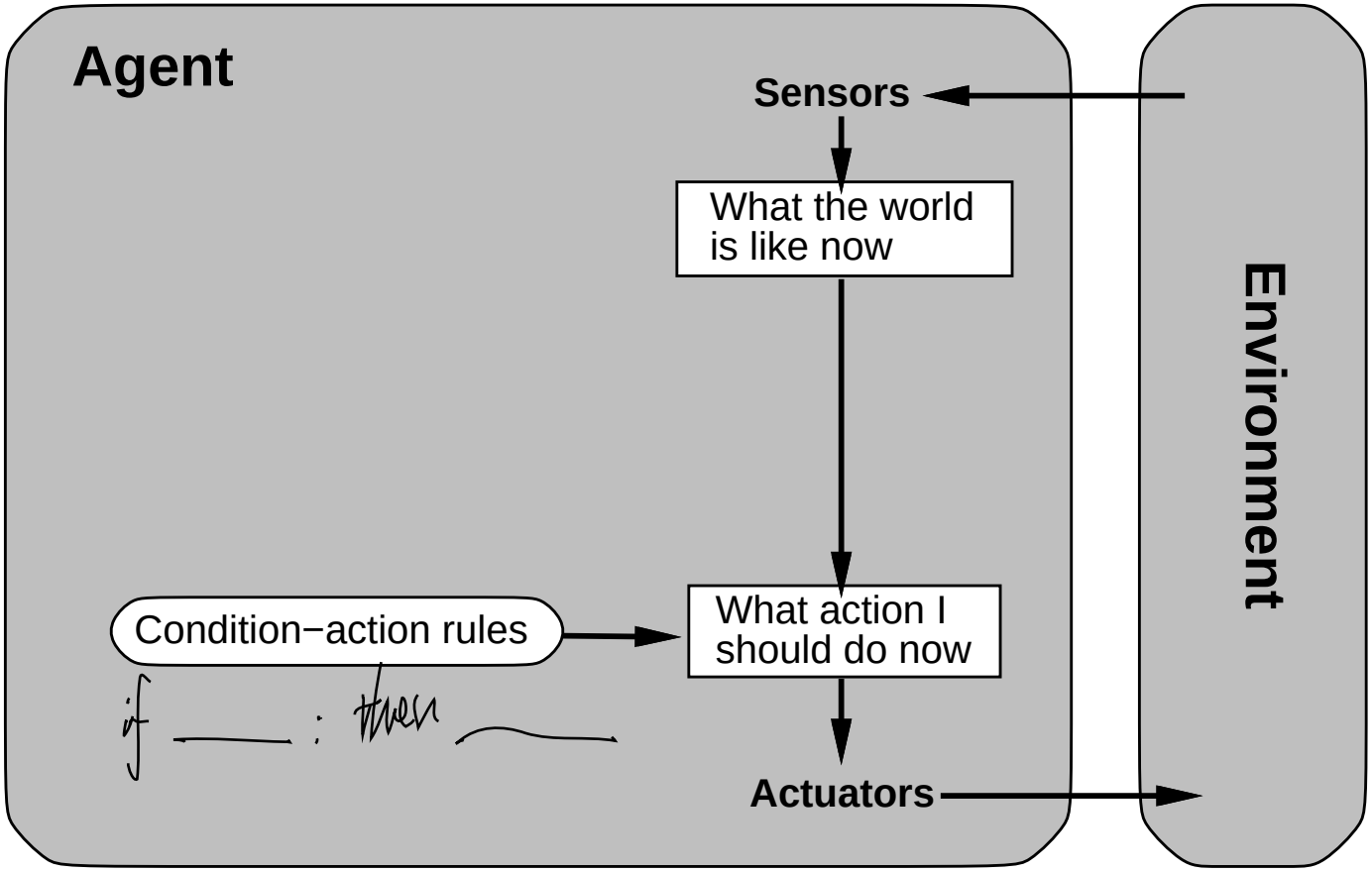
Agent types

Four basic types of agents 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



Decisions are made on the basis of the **current percept** only. Raises issues for partially observable environments

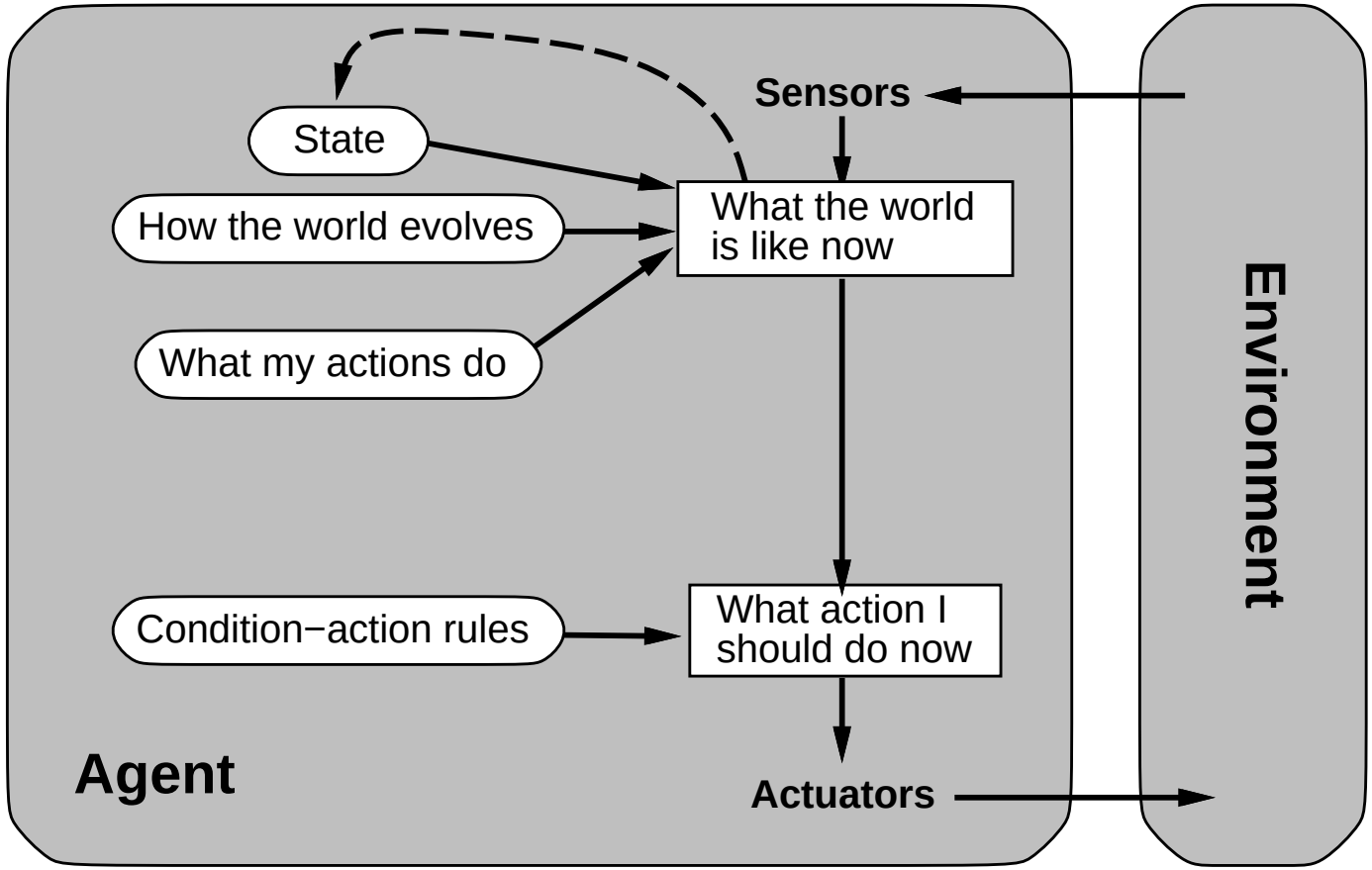
Example

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*

"model-based" agents

Reflex agents with state

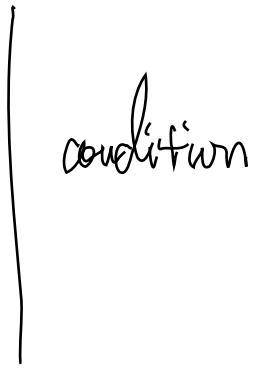


The internal state keeps track of relevant unobservable aspects of the environment. The environment model describes how the environment works (how the environment state is affected by actions)

Example

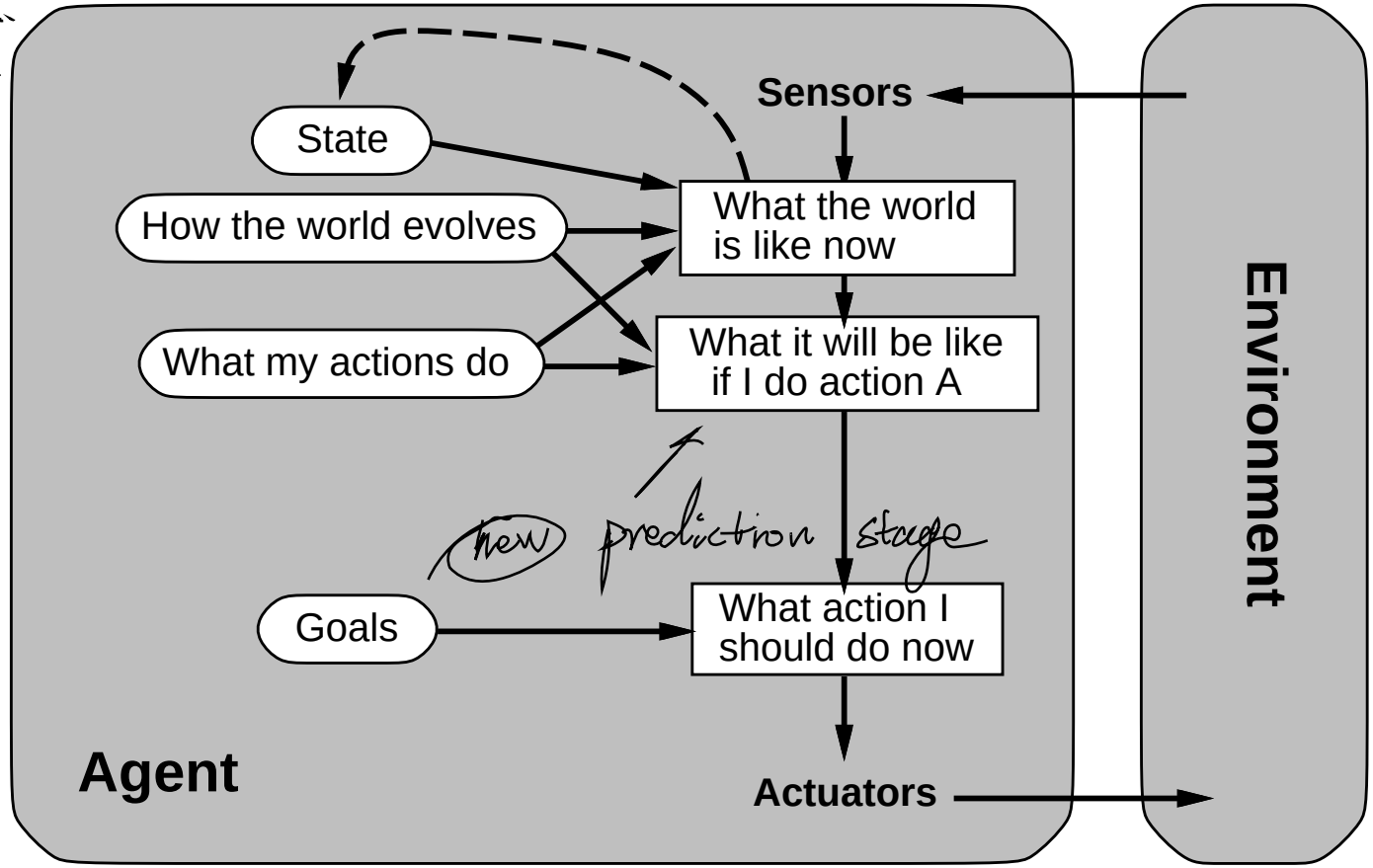
```
function VACUUM-AGENT-WITH-STATE( (location,status)) returns an action
static: last_A, last_B, numbers, initially ∞

    increment last_A and last_B
    if location = A then last_A = 0
    else last_B = 0
    case
        status = Dirty:
            return Suck
        location = A:
            if last_B > 3 then return Right
            else return NoOp
        location = B:
            if last_A > 3 then return Left
            else return NoOp
```



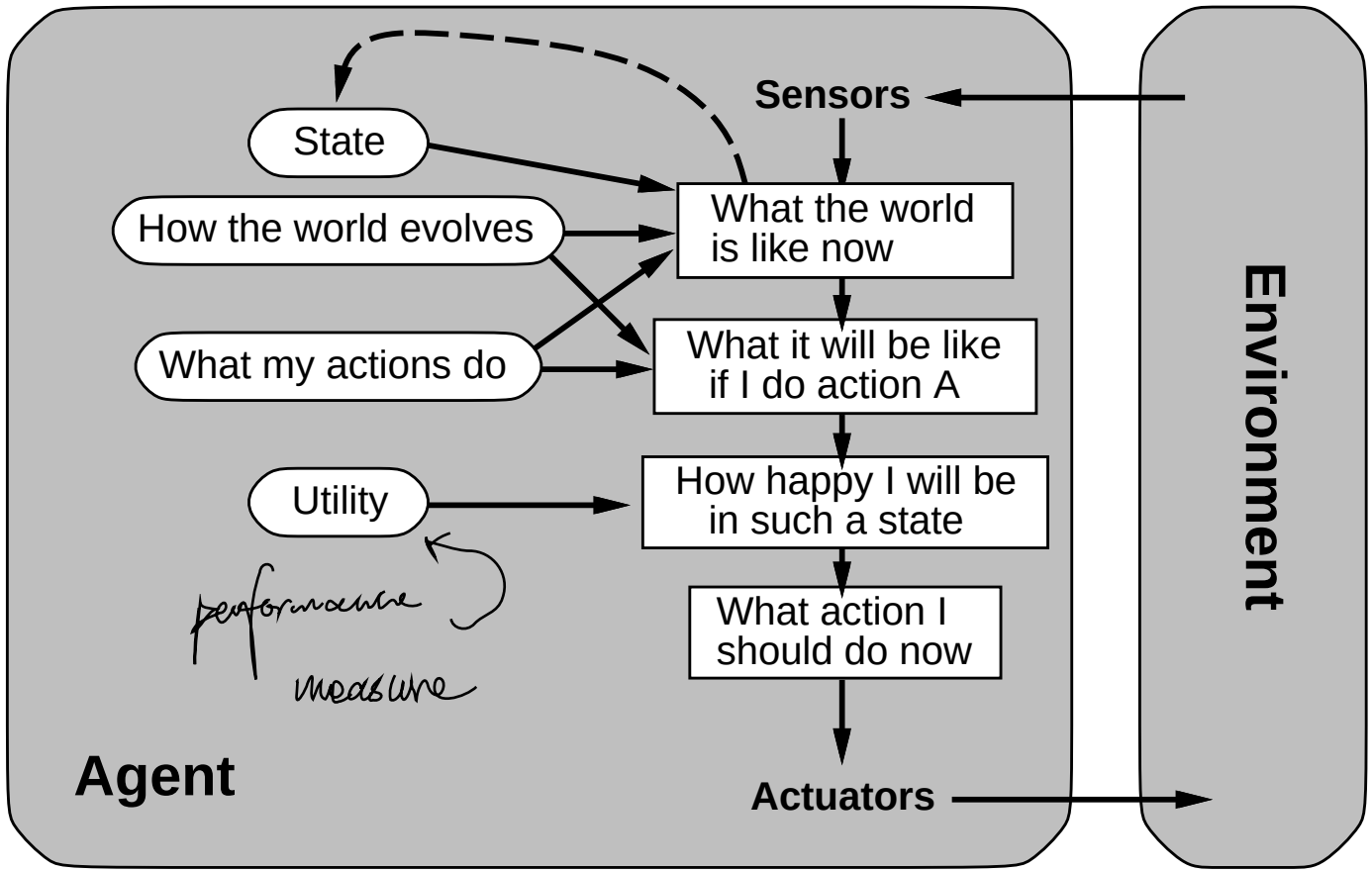
The time passed since a location was visited is a proxy for the likelihood of this location's status changing from clean to dirty.

Goal-based agents



The goal describes desirable situations. The agent combines goal and environment model to choose actions. Planning and search are AI subfields devoted to building goal-based agents.

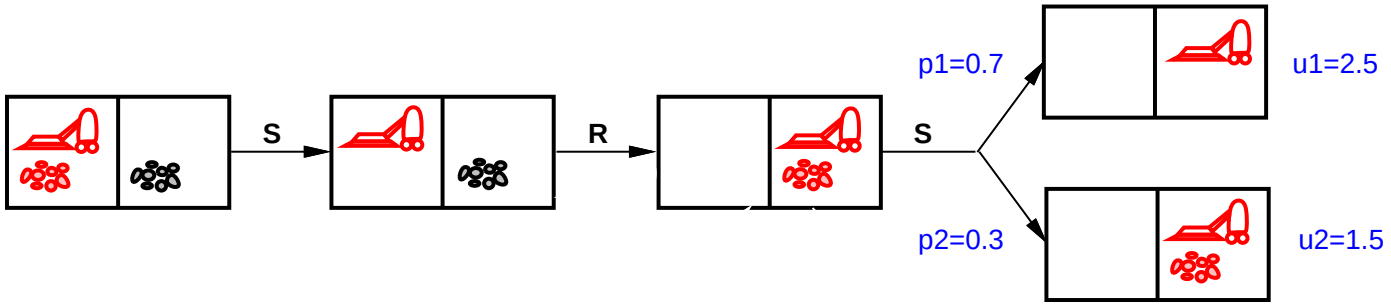
Utility-based agents



The utility function internalises the performance measure. Under uncertainty, the agent chooses actions that maximise the expected utility.

Utility-based agents

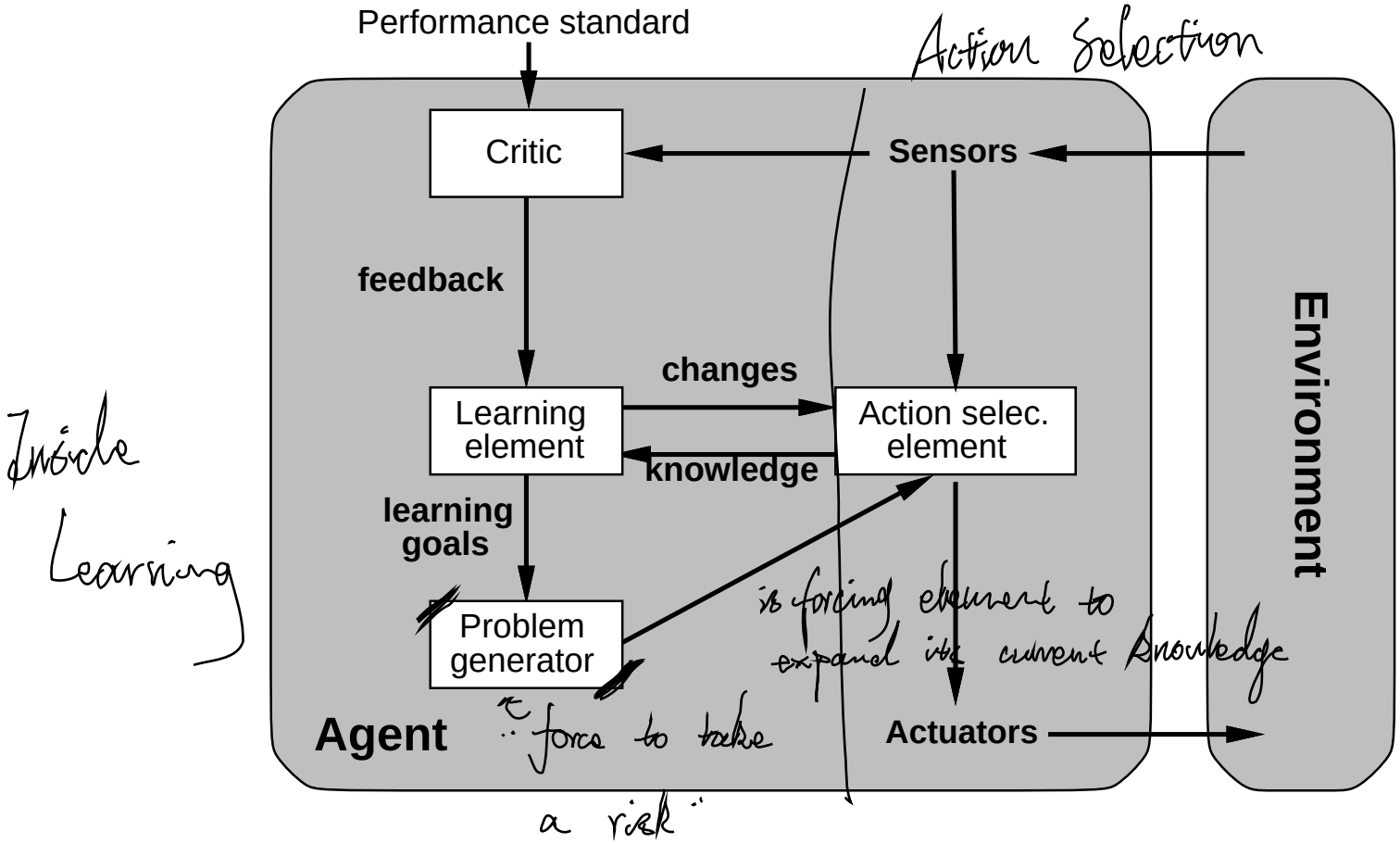
Rational agent: chooses the action that maximises expected utility:



Expected utility of $Suck$: $p1 \times u1 + p2 \times u2 = 0.7 \times 2.5 + 0.3 \times 1.5 = 2.2$

- $Suck$ has an expected utility of 2.2
- $NoOp$ has an expected utility of 2
- $Left$ has an expected utility of 1.5

Learning agents



The action selection element is what we described earlier. The learning element uses feedback from the critic to modify the action selection. The problem generator suggests actions that lead to new informative experience.

Exploration vs Exploitation

A fundamental dilemma for learning agents

♢ Exploitation: greedily uses what the agent has learnt to select the action that will, in the light of the current knowledge, have the best outcome

♢ Exploration: Taking some other (possibly random) action to learn more, hoping to find something even better than what is currently known

♢ In practice, agents must explore to avoid getting stuck in severely sub-optimal behavior, but exploration has a cost.

♢ Typically, a smart agent explores more in early stages than later on

Summary

Agents interact with environments through actuators and sensors

The agent function describes what the agent does in all circumstances

Agent programs implement agent functions

The performance measure evaluates the environment sequence

A perfectly rational agent maximizes expected performance

PEAS descriptions define task environments

Environments are categorized along several dimensions:
observable? deterministic? known? episodic?
static? discrete? single-agent?

Several basic agent architectures exist:
reflex, reflex with state, goal-based, utility-based

All agents can improve their performance through learning