SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

19AMB303-FULL STACK AI

M.POORNIMA DEVI,AP/AIML

Intelligent Agents

- What is an agent ?
  - An **agent** is anything that **perceiving** its environment through **sensors** and **acting** upon that environment through **actuators**
  - Example:
    - Human is an agent
    - A robot is also an agent with cameras and motors
    - A thermostat detecting room temperature.

Intelligent Agents

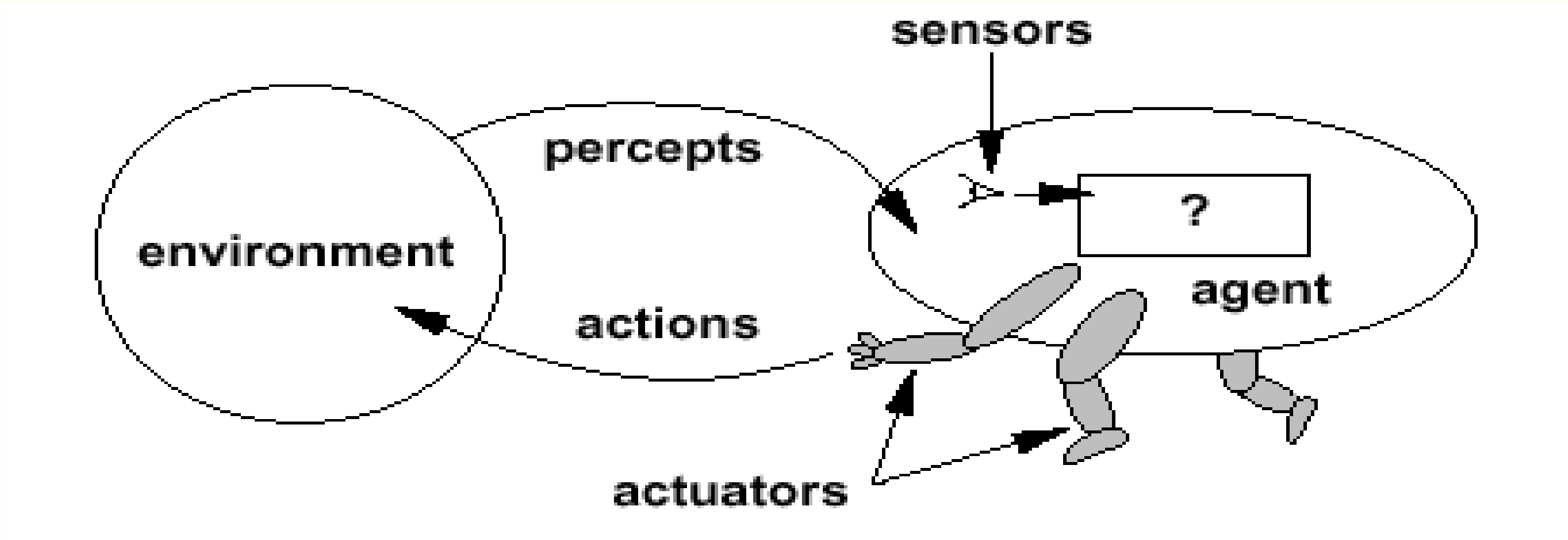
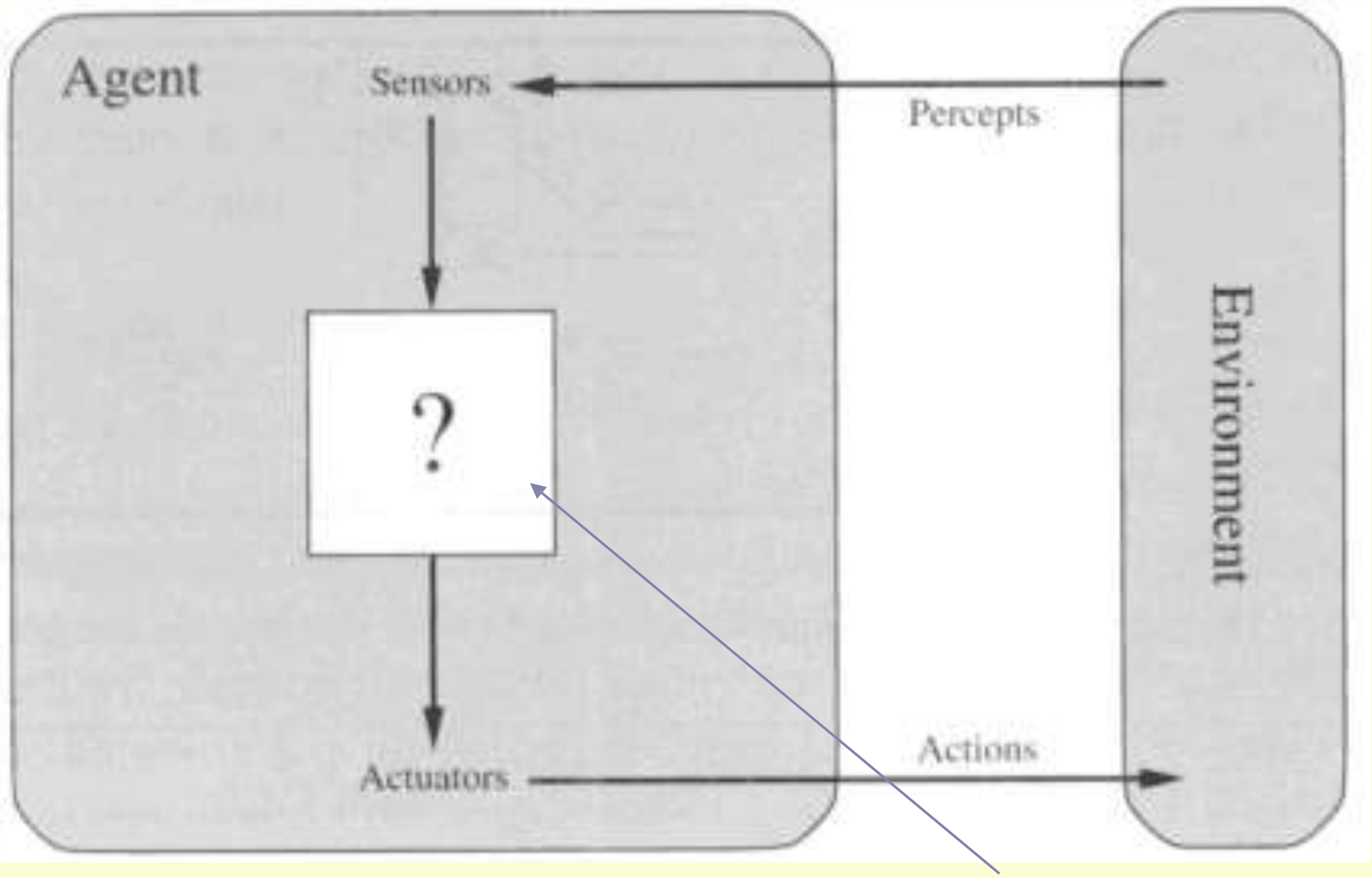


Diagram of an agent



What AI should fill

Simple Terms

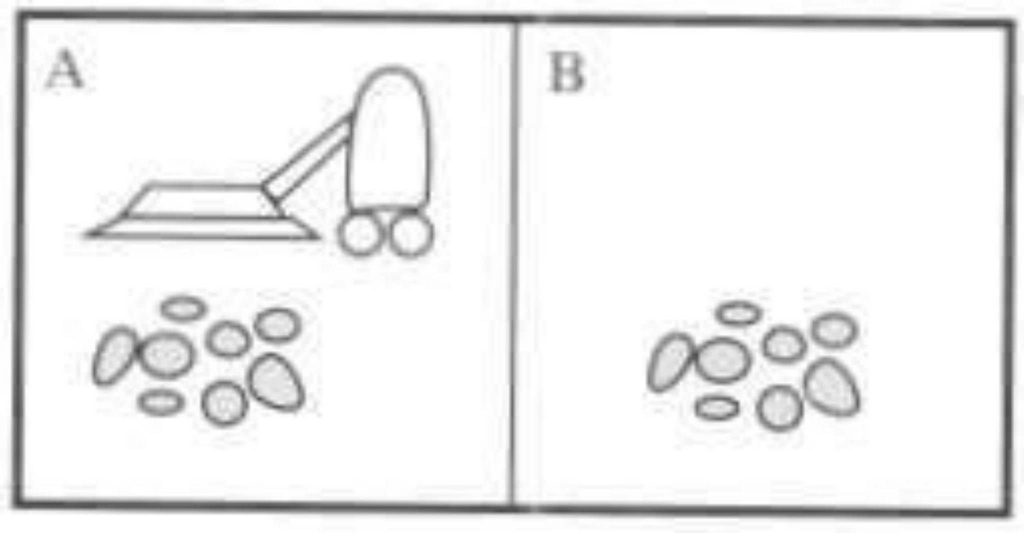
- Percept
  - Agent's perceptual inputs at any given instant
- Percept sequence
  - Complete history of everything that the agent has ever perceived.

Agent function & program

- Agent's behavior is *mathematically* described by
  - **Agent function**
  - A function mapping any given percept sequence to an action
- *Practically* it is described by
  - An **agent program**
  - The real implementation

Vacuum-cleaner world

- Perception: Clean or Dirty? where it is in?
- Actions: Move left, Move right, suck, do nothing



Vacuum-cleaner world

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.

Program implements the agent function tabulated in Fig. 2.3

Function Reflex-Vacuum-Agent([*location*,*status*])
return an action

If *status* = *Dirty* **then return** *Suck*

else if *location* = *A* **then return** *Right*

else if *location* = *B* **then return** *left*

Concept of Rationality

- Rational agent
  - One that does the right thing
  - = every entry in the table for the agent function is correct (rational).
- What is correct?
  - The actions that cause the agent to be most successful
  - So we need ways to measure success.

Performance measure

- Performance measure
  - An objective function that determines
    - How the agent does successfully
    - E.g., 90% or 30% ?
- An agent, based on its percepts
  - → action sequence :

  if desirable, it is said to be performing well.
  - No universal performance measure for all agents

Performance measure

- A general rule:
  - Design performance measures according to
    - What one actually wants in the environment
    - Rather than how one thinks the agent should behave
- E.g., in vacuum-cleaner world
  - We want the floor clean, no matter how the agent behave
  - We don’t restrict how the agent behaves

Rationality

- What is rational at any given time depends on four things:
  - The performance measure defining the criterion of success
  - The agent’s prior knowledge of the environment
  - The actions that the agent can perform
  - The agents’s percept sequence to date

Rational agent

- For each possible percept sequence,
  - an rational agent should select
    - an action expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has
- E.g., an exam
  - Maximize marks, based on

the questions on the paper & your knowledge

Rational agent

- Performance measure
  - Awards one point for each clean square
    - at each time step, over 10000 time steps
- Prior knowledge about the environment
  - The geography of the environment
  - Only two squares
  - The *effect* of the actions

Example of a rational agent

- Actions that can perform
  - Left, Right, Suck and NoOp
- Percept sequences
  - Where is the agent?
  - Whether the location contains dirt?

- Under this circumstance, the agent is rational.

Omniscience

- An omniscient agent
  - Knows the *actual* outcome of its actions in advance
  - No other possible outcomes
  - However, impossible in real world
- An example
  - crossing a street but died of the fallen cargo door from 33,000ft → irrational?

Omniscience

- Based on the circumstance, it is rational.
- As rationality maximizes
  - *Expected* performance
- Perfection maximizes
  - *Actual* performance
- Hence rational agents are not *omniscient*.

Learning

- Does a rational agent depend on only current percept?
  - No, the past percept sequence should also be used
  - This is called **learning**
  - After experiencing an episode, the agent
    - should adjust its behaviors to perform better for the same job next time.

Autonomy

- If an agent just relies on the prior knowledge of its designer rather than its own percepts then the agent lacks ***autonomy***

A rational agent should be autonomous- it should learn what it can to compensate for partial or incorrect prior knowledge.

Software Agents

- Sometimes, the environment may not be the real world
  - E.g., flight simulator, video games, Internet
  - They are all artificial but very complex environments
  - Those agents working in these environments are called
    - Software agent (softbots)
    - Because all parts of the agent are software

Task environments

- Task environments are the problems
  - While the rational agents are the solutions
- Specifying the task environment
  - PEAS description as fully as possible
    - Performance
    - Environment
    - Actuators
    - Sensors

    In designing an agent, the first step must always be to specify the task environment as fully as possible.
- Use automated taxi driver as an example

Task environments

- Performance measure
  - How can we judge the automated driver?
  - Which factors are considered?
    - getting to the correct destination
    - minimizing fuel consumption
    - minimizing the trip time and/or cost
    - minimizing the violations of traffic laws
    - maximizing the safety and comfort, etc.

THANKYOU

SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

19AMB303-FULL STACK AI

M.POORNIMA DEVI,AP/AIML

Types of agent programs

- Four types
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based agents

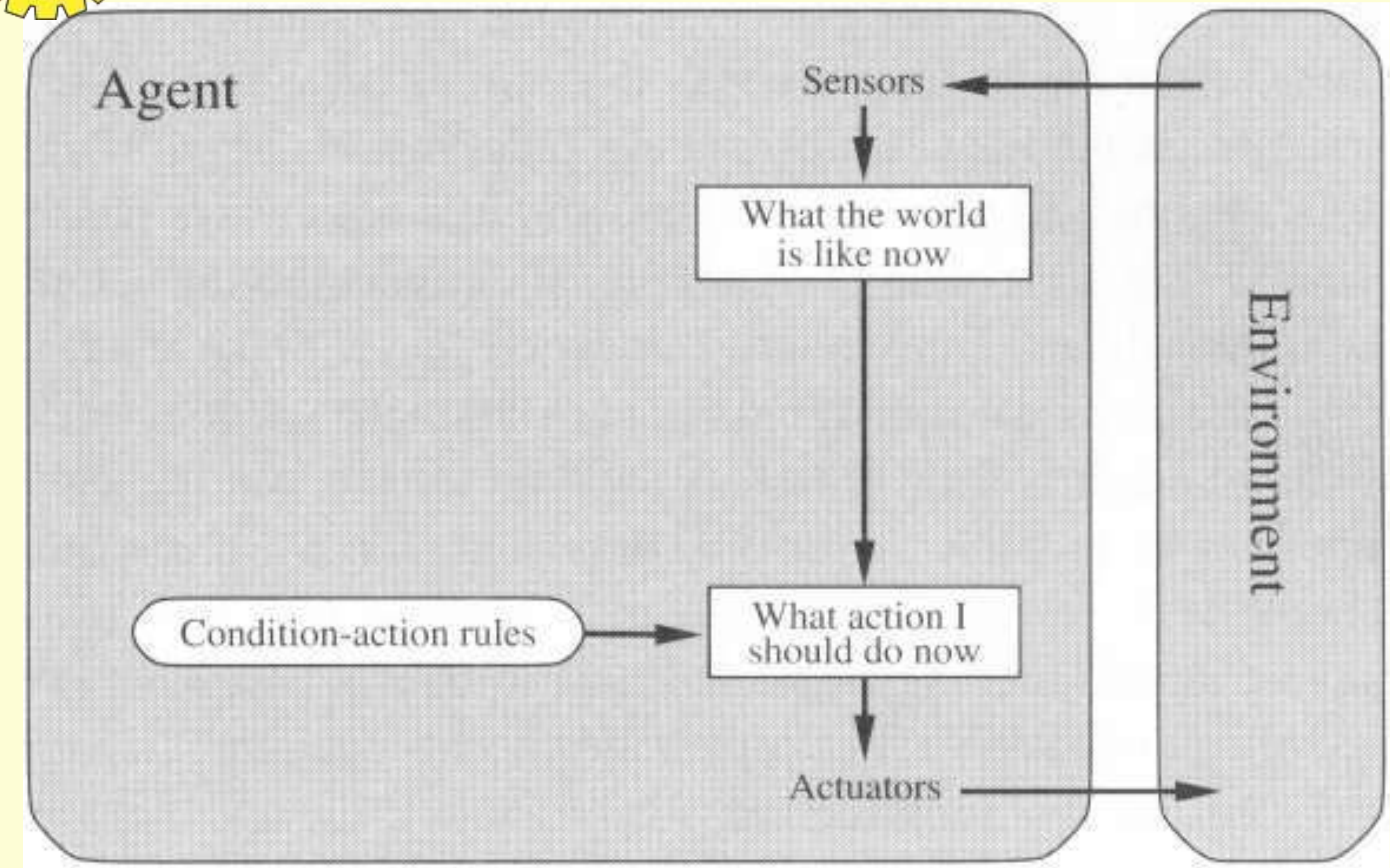
Simple reflex agents

- It uses just ***condition-action rules***
  - The rules are like the form “if …   then … ”
  - efficient but have narrow range of applicability
  - Because knowledge sometimes cannot be stated explicitly
  - Work only
    - if the environment is fully observable

Simple reflex agents

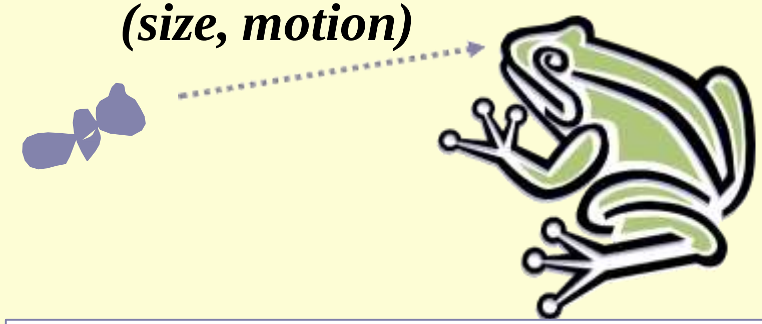
```
function SIMPLE-REFLEX-AGENT(percept) returns action
   static: rules, a set of condition-action rules

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

Simple reflex agents (2)



A Simple Reflex Agent in Nature

percepts
(size, motion)



<u>RULES:</u>

(1) If small moving object,
then activate SNAP

(2) If large moving object,
then activate AVOID and inhibit SNAP

ELSE (not moving) then NOOP

needed for completeness

Action: SNAP or AVOID or NOOP

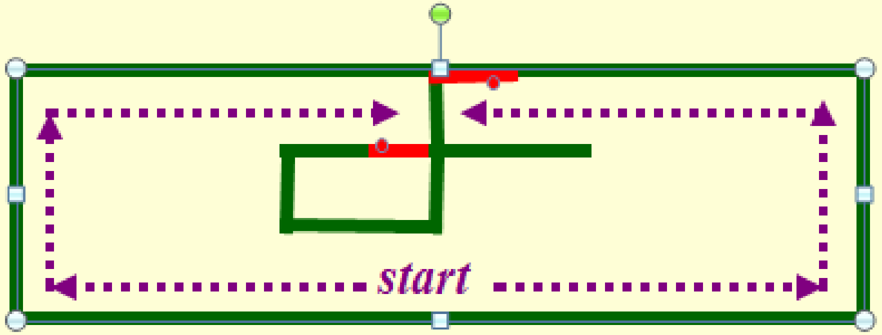
Model-based Reflex Agents

- For the world that is partially observable
  - the agent has to keep track of an internal state
    - That depends on the percept history
    - Reflecting some of the unobserved aspects
    - E.g., driving a car and changing lane
- Requiring two types of knowledge
  - How the world evolves independently of the agent
  - How the agent's actions affect the world

Example Table Agent With Internal State

IF	THEN
Saw an object ahead, and turned right, and it's now clear ahead	Go straight
Saw an object Ahead, turned right, and object ahead again	Halt
See no objects ahead	Go straight
See an object ahead	Turn randomly

Example Reflex Agent With Internal State: **Wall-Following**



Actions: left, right, straight, open-door

Rules:

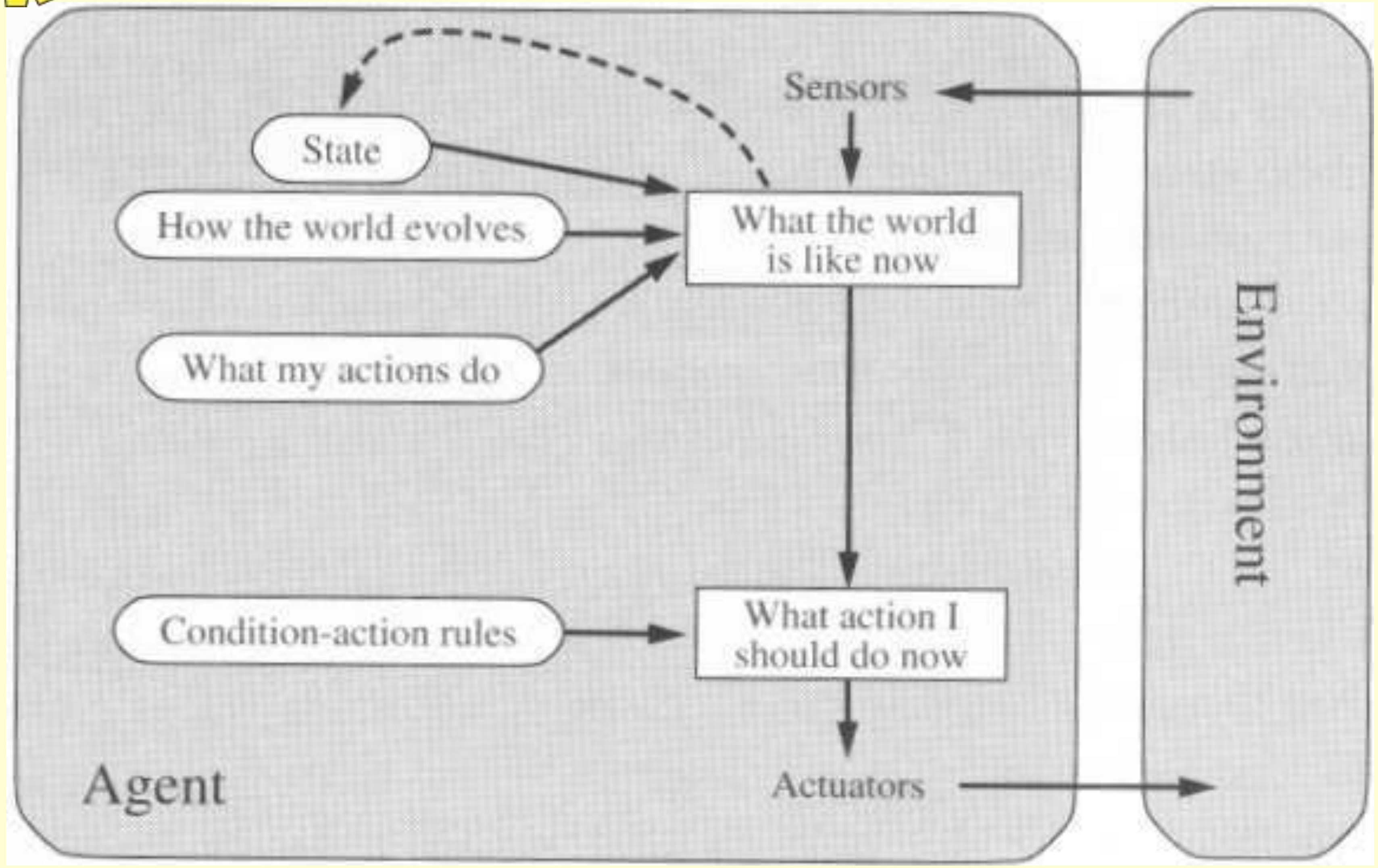
1. If open(left) & open(right) and open(straight) then choose randomly between right and left
2. If wall(left) and open(right) and open(straight) then straight
3. If wall(right) and open(left) and open(straight) then straight
4. If wall(right) and open(left) and wall(straight) then left
5. If wall(left) and open(right) and wall(straight) then right
6. If wall(left) and door(right) and wall(straight) then open-door
7. If wall(right) and wall(left) and open(straight) then straight.
8. (Default) Move randomly

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

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

The agent is with memory

Model-based Reflex Agents



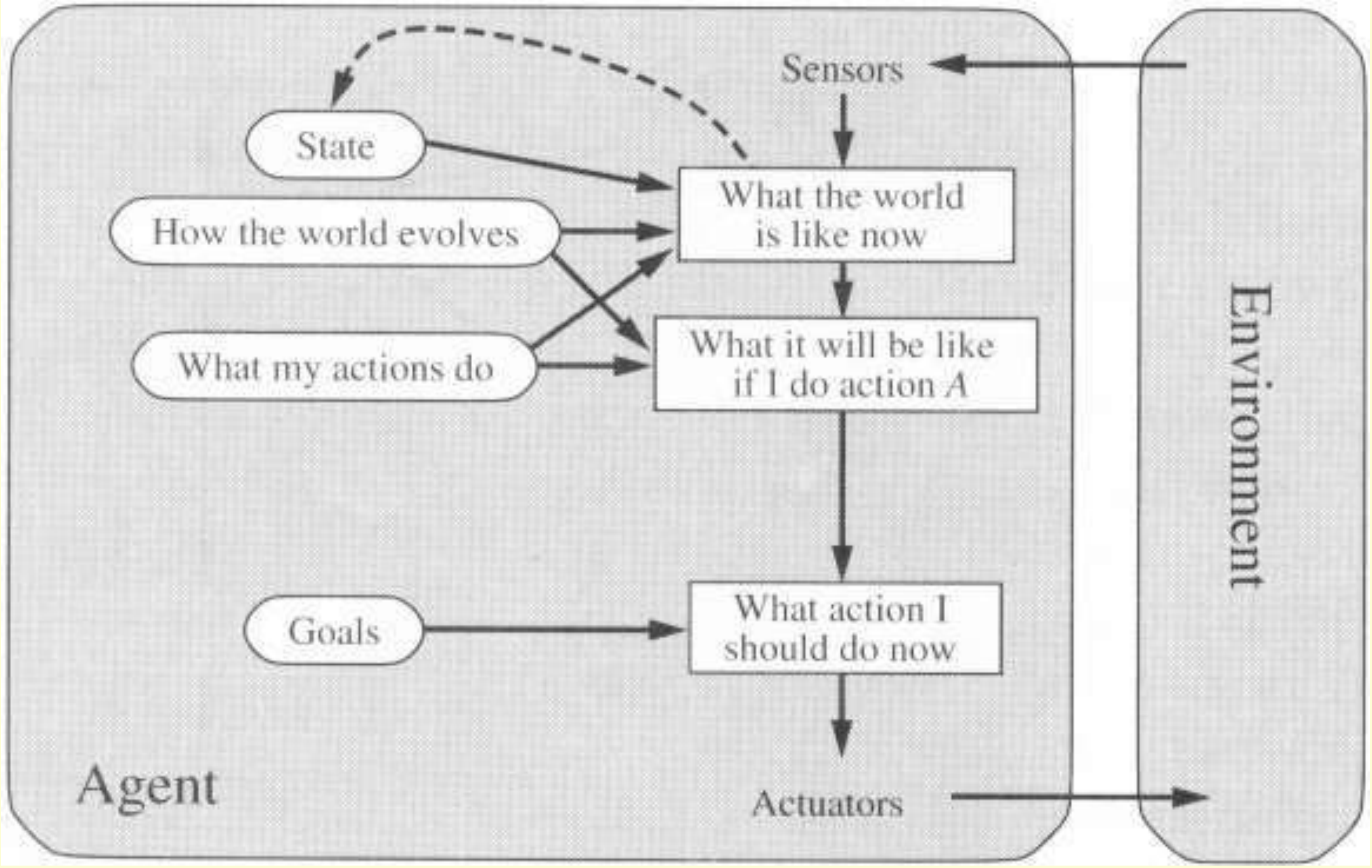
Goal-based agents

- Current state of the environment is always not enough
- The goal is another issue to achieve
  - Judgment of rationality / correctness
- Actions chosen → goals, based on
  - the current state
  - the current percept

Goal-based agents

- Conclusion
  - Goal-based agents are less efficient
  - but more flexible
    - Agent ← Different goals ← different tasks
  - Search and planning
    - two other sub-fields in AI
    - to find out the action sequences to achieve its goal

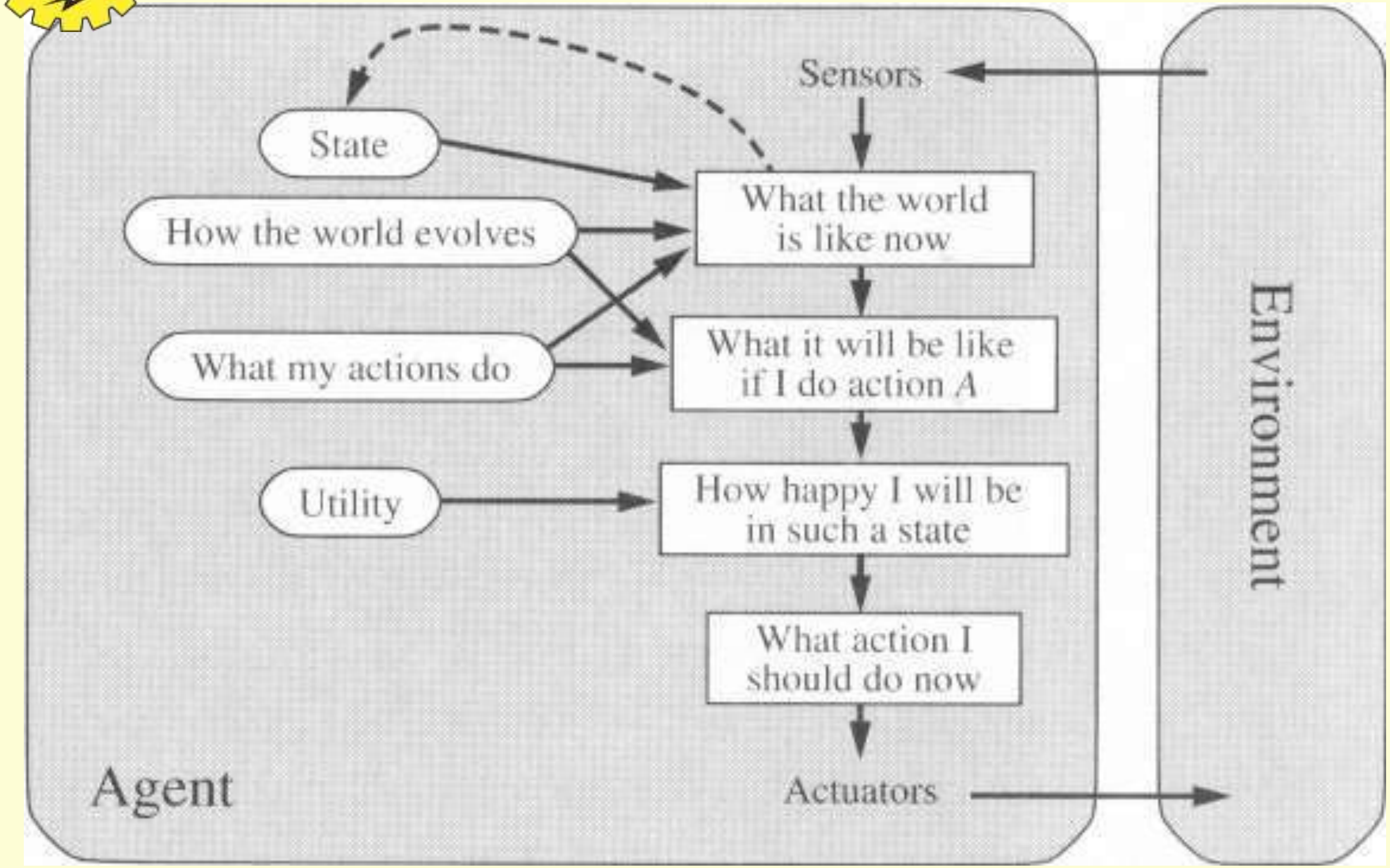
Goal-based agents



Utility-based agents

- Goals alone are not enough
  - to generate **high-quality** behavior
  - E.g. meals in Canteen, good or not ?
- Many action sequences → the goals
  - some are better and some worse
  - If goal means success,
  - then **utility** means the degree of success (how successful it is)

Utility-based agents (4)



Utility-based agents

- it is said state A has higher utility
  - If state A is more preferred than others
- Utility is therefore a function
  - that maps a state onto a real number
  - the degree of success

Utility-based agents (3)

- Utility has several advantages:
  - When there are conflicting goals,
    - Only some of the goals but not all can be achieved
    - utility describes the appropriate trade-off
  - When there are several goals
    - None of them are achieved **certainly**
    - utility provides a way for the decision-making

Learning Agents

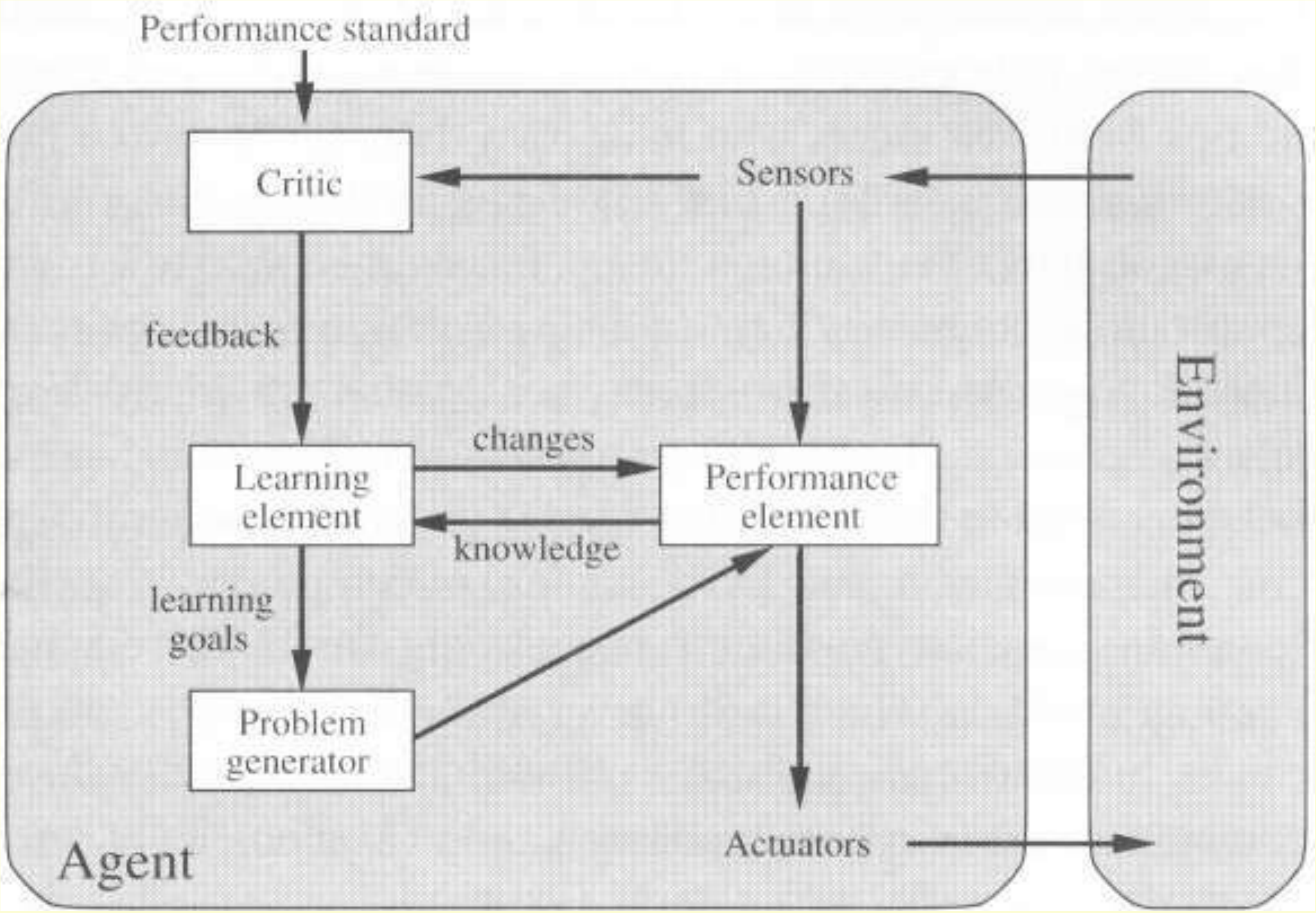
- After an agent is programmed, can it work immediately?
  - No, it still need teaching
- In AI,
  - Once an agent is done
  - We teach it by giving it a set of examples
  - Test it by using another set of examples
- We then say the agent learns
  - A learning agent

Learning Agents

- Four conceptual components
  - Learning element
    - Making improvement
  - Performance element
    - Selecting external actions
  - Critic
    - Tells the Learning element how well the agent is doing with respect to fixed performance standard.

    (Feedback from user or examples, good or not?)
  - Problem generator
    - Suggest actions that will lead to new and informative experiences.

Learning Agents



THANKYOU

SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

19AMB303-FULL STACK AI

M.POORNIMA DEVI,AP/AIML

Problem-solving agent

- Problem-solving agent
  - A kind of goal-based agent
  - It solves problem by
    - finding sequences of actions that lead to desirable states (goals)
  - To solve a problem,
    - the first step is the ***goal formulation***, based on the current situation

Goal formulation

- The goal is formulated
  - as a set of states, in which the goal is satisfied
- Reaching from initial state → goal state
  - Actions are required
- *Goal formulation, based on the current situation and the agent's performance measure, is the first step in problem solving.*
- Actions are the operators
  - causing transitions between states
  - **Actions** should be abstract enough at a certain degree, instead of very detailed
  - E.g., turn left VS turn left 30 degree, etc.

Problem formulation

- The process of deciding
  - what actions and states to consider, given a goal
- E.g., driving Amman → Zarqa
  - in-between states and actions defined
  - States: Some places in Amman & Zarqa
  - Actions: Turn left, Turn right, go straight, accelerate & brake, etc.

Search

- Because there are many ways to achieve the same goal
  - Those ways are together expressed as a tree
  - Multiple options of unknown value at a point,
    - the agent can examine different possible sequences of actions, and choose the best
  - This process of looking for such a sequence is called ***search***
  - A search algorithm takes a problem as input and returns a ***solution*** *in the form of an action sequence.*

Search algorithm

- Defined as
  - taking a *problem*
  - and returns a *solution*
- Once a solution is found
  - the agent follows the solution
  - and carries out the list of actions – execution phase
- Design of an agent
  - "Formulate, search, execute"

A simple problem-solving agent

```
function SIMPLE-PROBLEM-SOLVING-AGENT(p) returns an action
    inputs: p, a percept
    static: s, an action sequence, initially empty
            state, some description of the current world state
            g, a goal, initially null
            problem, a problem formulation
    state ← UPDATE-STATE(state, p)
    if s is empty then
        g ← FORMULATE-GOAL(state)
        problem ← FORMULATE-PROBLEM(state, g)
        s ← SEARCH(problem)
    action ← RECOMMENDATION(s, state)
    s ← REMAINDER(s, state)
    return action
```

Well-defined problems and solutions

- A problem is defined by 4 components:
  - The ***initial state***
    - that the agent starts in
  - The set of possible **actions**

Transition model: *description of what each action does.*

(***successor functions***): refer to any state reachable from given state by a single action

Initial state, actions and Transition model define the ***state space***

- the set of all states reachable from the initial state by any sequence of actions.

A ***path*** in the state space:

- any sequence of states connected by a sequence of actions.

Well-defined problems and solutions

- The ***goal test***
  - Applied to the current state to test
    - if the agent is in its goal

-Sometimes there is an explicit set of possible goal states. (example: in Amman).

-Sometimes the goal is described by the properties

- instead of stating explicitly the set of states

Example: Chess

- the agent wins if it can capture the KING of the opponent on next move (checkmate).
- no matter what the opponent does

Well-defined problems and solutions

- A ***path cost*** function,
  - assigns a numeric cost to each path
  - = performance measure
  - denoted by ***g***
  - to distinguish the best path from others
- Usually the path cost is
  - the sum of the **step costs** of the individual actions (in the action list)

Well-defined problems and solutions

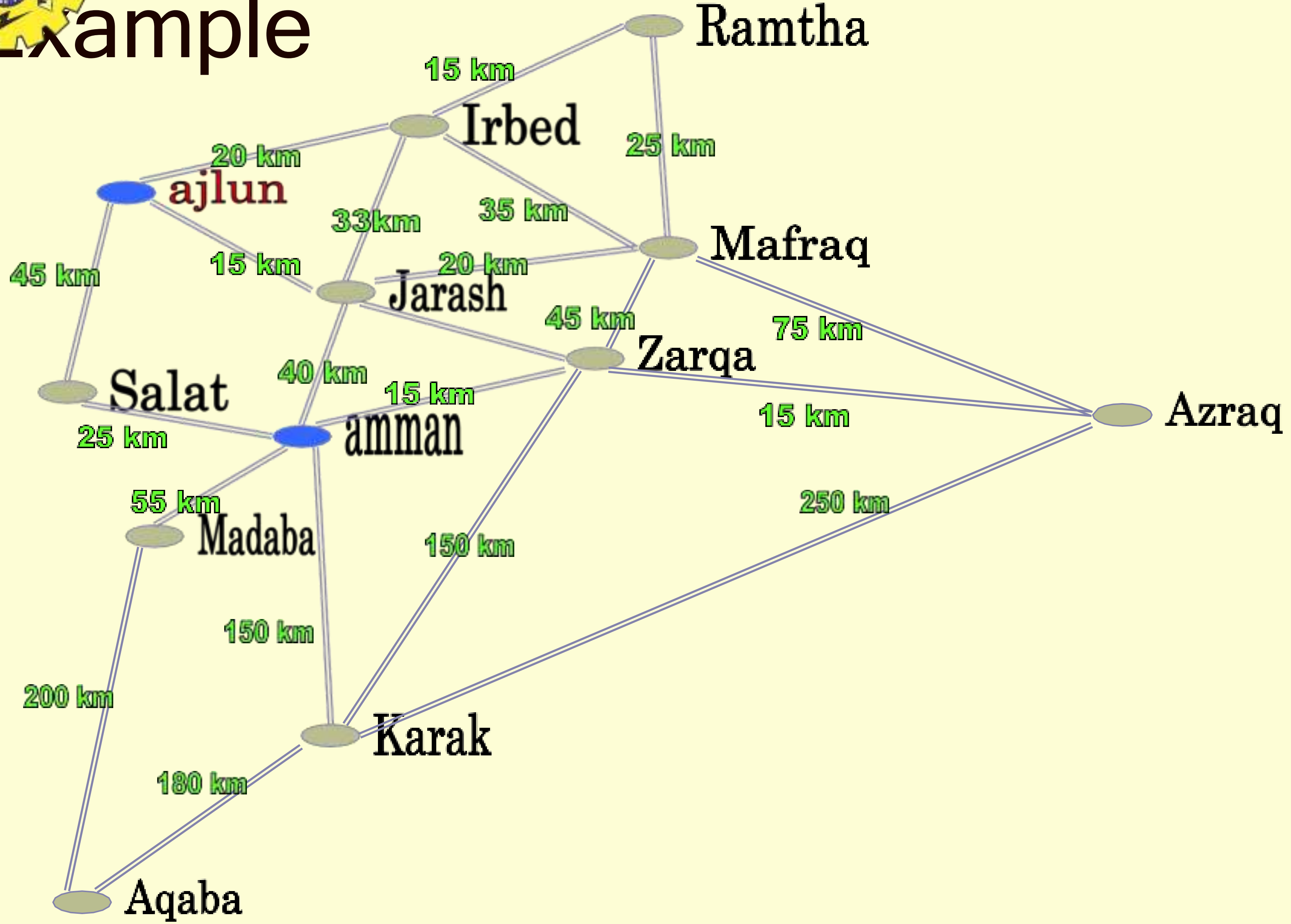
- Together a problem is defined by
  - Initial state
  - Actions
  - Successor function
  - Goal test
  - Path cost function
- The ***solution*** of a problem is then
  - *a path from the initial state to a state satisfying the goal test*
- ***Optimal*** solution
  - the solution with lowest path cost among all solutions

Formulating problems

- Besides the four components for problem formulation
  - anything else?
- ***Abstraction***
  - the process to take out the irrelevant information
  - leave the most essential parts to the description of the states

  (Remove detail from representation)
  - **Conclusion**: Only the most important parts *that are contributing to searching* are used

Example



Ramtha
15 km
Irbed
25 km
20 km
ajlun
33km
35 km
Mafraq
45 km
15 km
20 km
Jarash
45 km
Zarqa
75 km
40 km
Salat
15 km
amman
15 km
Azraq
25 km
55 km
Madaba
250 km
150 km
150 km
200 km
Karak
180 km
Aqaba

From our Example

1. Formulate Goal

   - Be In Amman

2. Formulate Problem

   - States : Cities
   - actions : Drive Between Cities

3. Find Solution

   - Sequence of Cities : ajlun – Jarash - Amman

Our Example

1. Problem : To Go from Ajlun to Amman

2. Initial State : Ajlun

3. Operator : Go from One City To another .

4. State Space : {Jarash , Salat , irbed,……..}

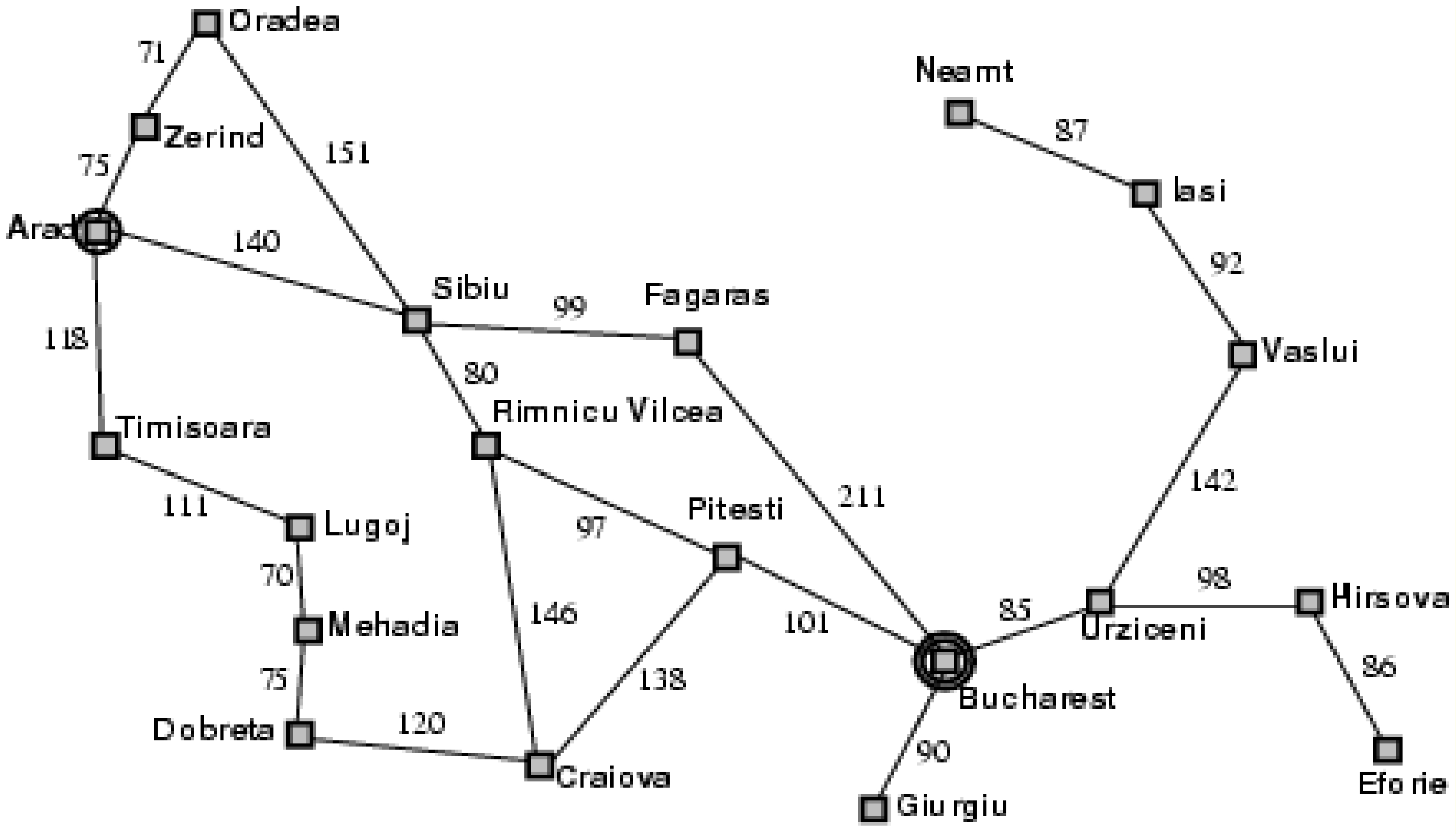
5. Goal Test : are the agent in Amman.

6. Path Cost Function : Get The Cost From The Map.

7. Solution : { {Aj → Ja → Ir → Ma → Za → Am} , {Aj →Ir → Ma → Za → Am} … . {Aj → Ja → Am} }

8. State Set Space : {Ajlun → Jarash → Amman}

Example: Romania

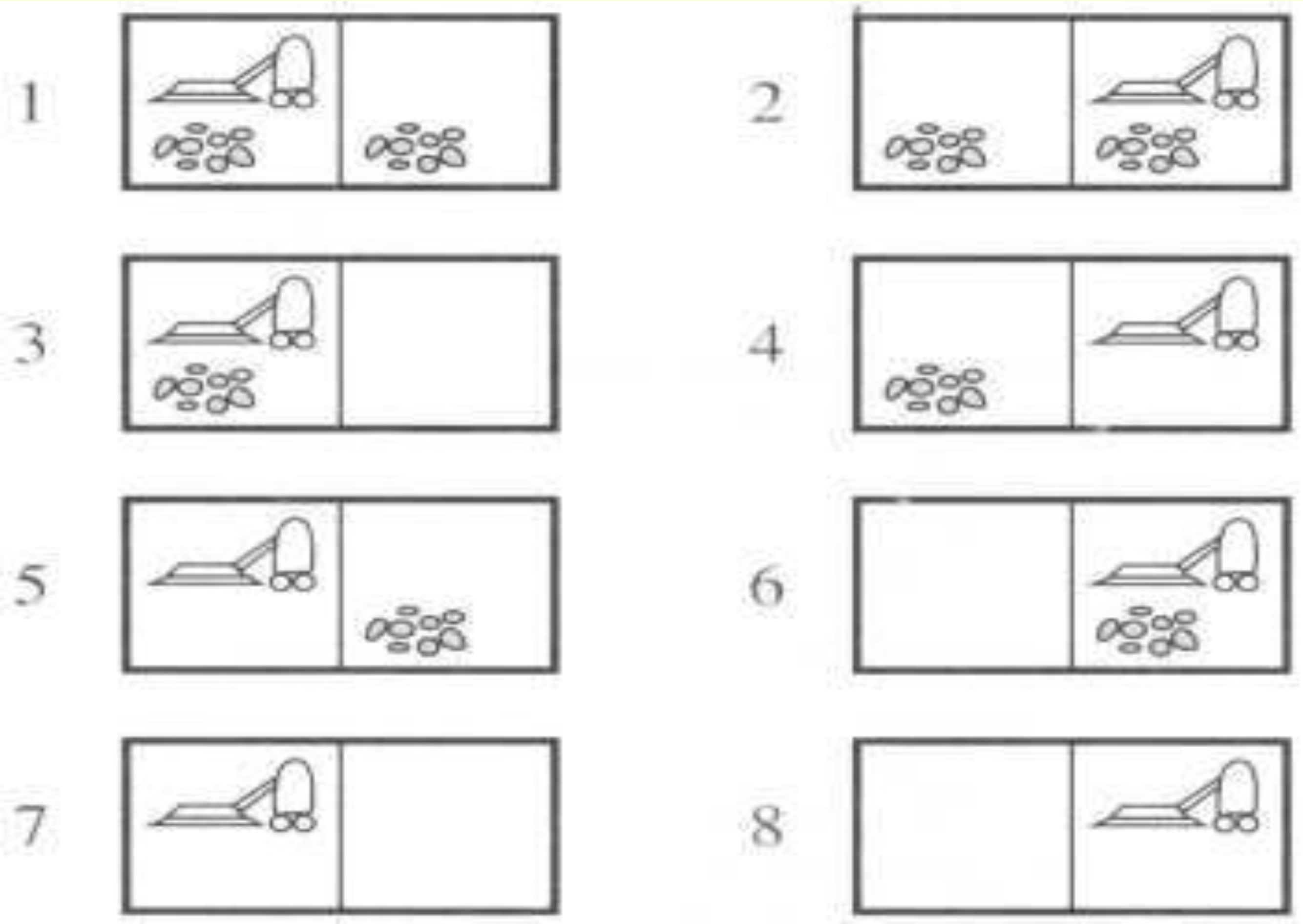


Example problems

- *Toy problems*
  - those intended to illustrate or exercise various problem-solving methods
  - E.g., puzzle, chess, etc.
- *Real-world problems*
  - tend to be more difficult and whose solutions people actually care about
  - E.g., Design, planning, etc.

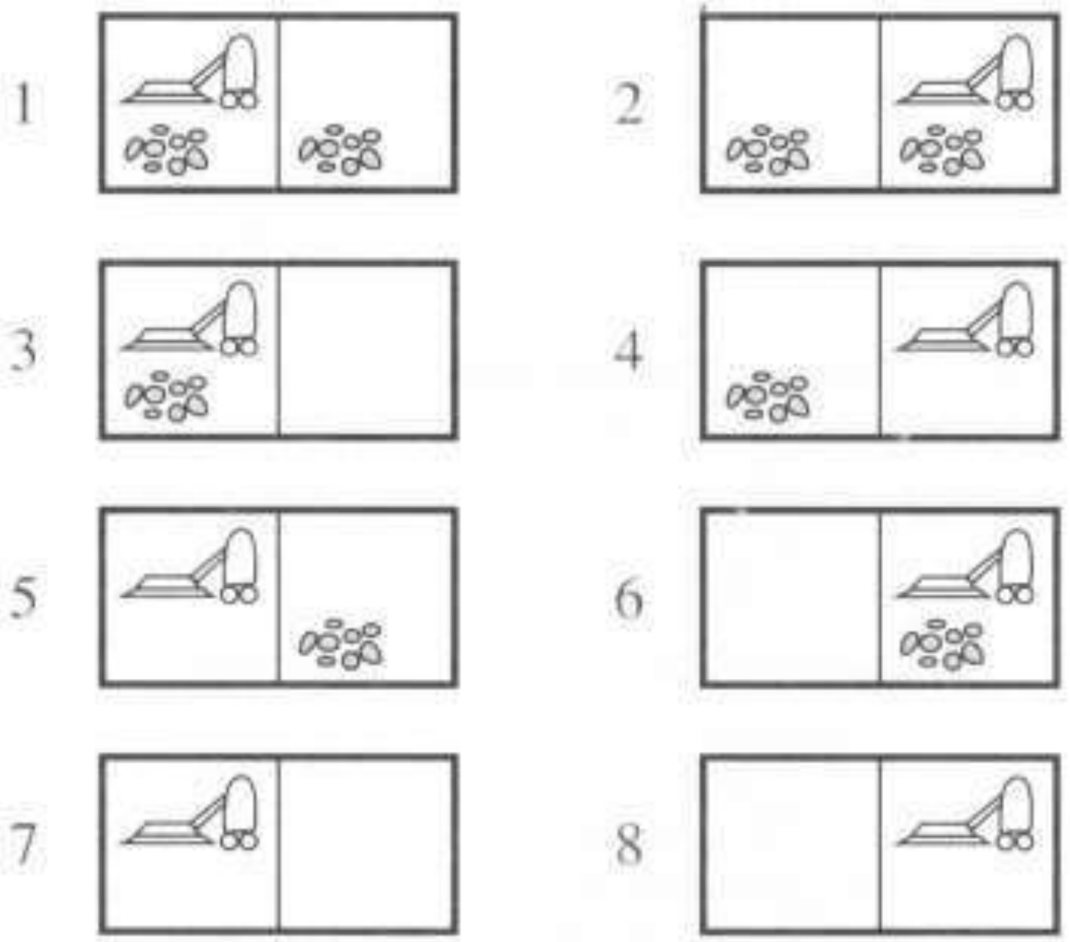
Toy problems

- Example: vacuum world

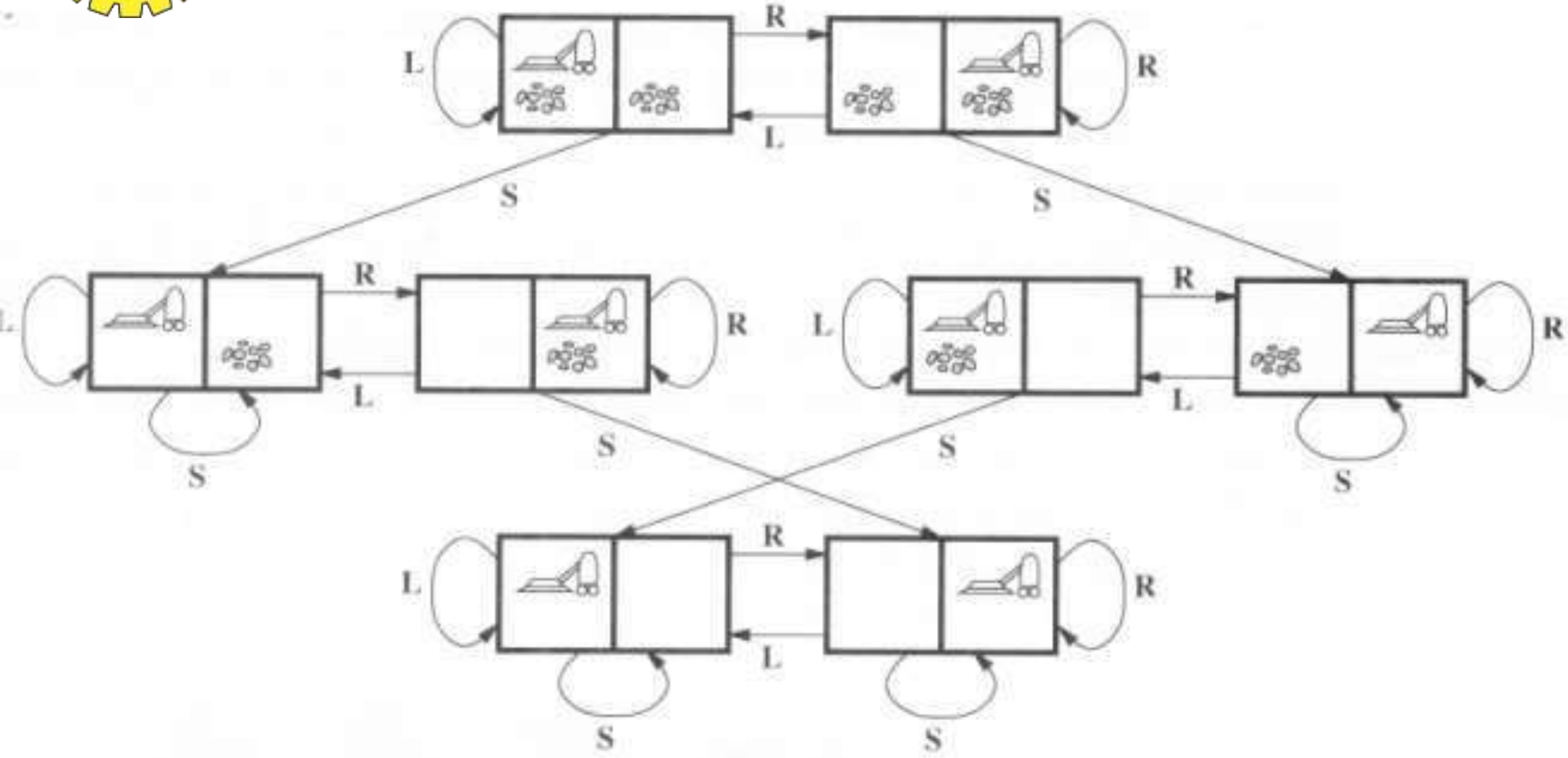


Toy problems

- Example: vacuum world



- **Number of states:** 8
- **Initial state:** Any
- **Number of actions:** 4
  - *left, right, suck, noOp*
- **Goal:** clean up all dirt
  - Goal states: {7, 8}
- **Path Cost:**
  - Each step costs 1



R
L
R
L
S
S
R
L
R
R
L
R
L
S
S
S
S
S
R
L
R
L
S
S

THANKYOU