Agent, Search and Game Playing

Topic Covered

1. Black- Box Model of Agent

2. Intentionality and Goals

3. Games, Search, Heuristic and Pruning

4. Strategies Rules

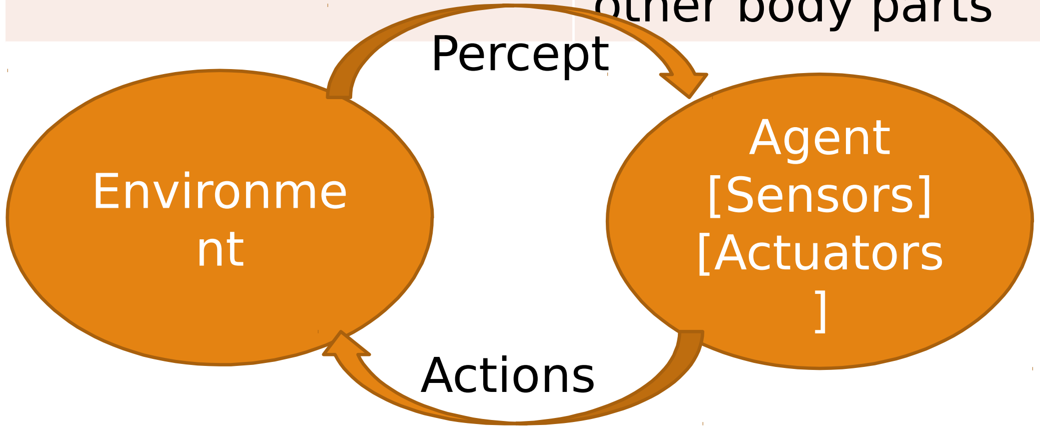
5. Making Simple Game- Playing Agent for TTT

6. Evaluation Functions, Utilitarian, Decision Making, Planning, Internal Representation

1. Black-Box Model of Agent

An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators.

Agent	**Human**	**Machine**
Sensors	Eyes, ears, other organs	Cameras, IR finder
Actuators	Hands, legs mouth, other body parts	Various motors for actuators



Agent = architecture + programme

Agent

Rational Agent

- Striving to do the right thing based on what it perceive and the action it can perform

-Performance Measure: An objective Criterion for success of an agent’s behaviour.

Ex: Vacuum Cleaner : - amount of dirt cleaned, amount of time consumed, amount of electricity consumed, amount of noise generated, etc.

Intelligent Agent : Self Driving Car:

PEAS(Performance, Environment, Actuator, Sensors)

P: Safe, Fast, Legal, comfortable trip, maximize profit

E: Road, Other Traffics, Pedestrians, Customers

A: Steering Wheels, accelerator, brake, signal, horn

S: Cameras, Sonar, Speedometer, GPS, Odometer, Engine sensors, keyboard

Agent

Types of Environment

Fully observable (vs. partially observable): An agent's sensors give it access to the complete state of the environment at each point in time.

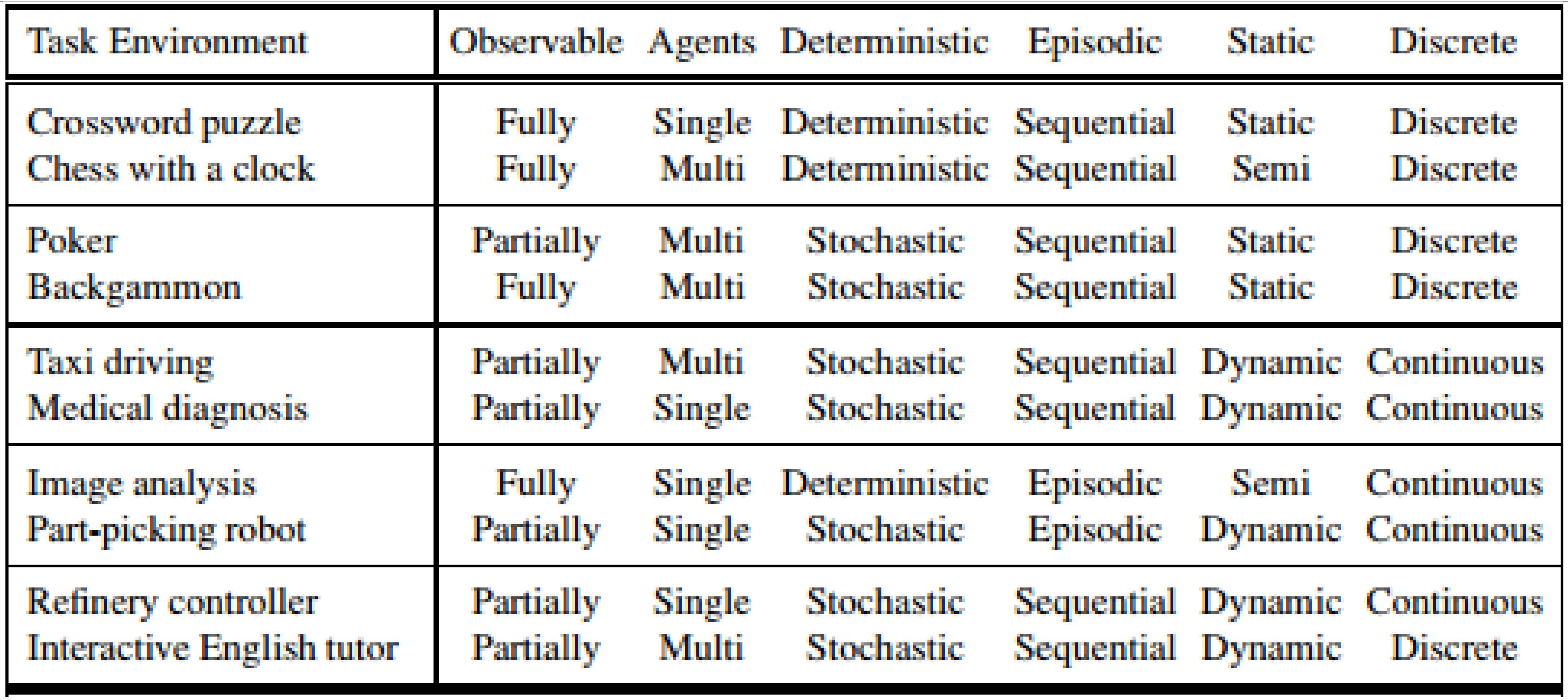
Deterministic (vs. stochastic): The next state of the environment is completely determined by the current state and the action executed by the agent. (If the environment is deterministic except for the actions of other agents, then the environment is strategic)

Episodic (vs. sequential): The agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself.

Static (vs. dynamic): The environment is unchanged while an agent is deliberating. (The environment is semi-dynamic if the environment itself does not change with the passage of time but the agent's performance score does)

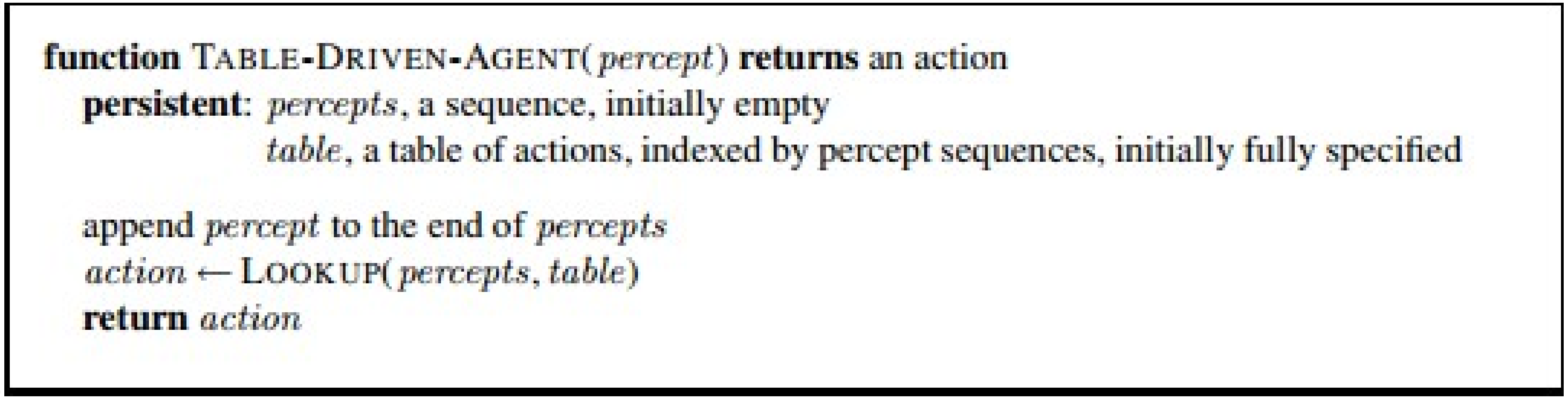
Discrete (vs. continuous): A limited number of distinct, clearly defined percepts and actions.

Single agent (vs. multi-agent): An agent operating by itself in an environment.



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 Programme: takes the current percept as input from the sensors and return an action to the actuators.



```
function TABLE-DRIVEN-AGENT(percept) returns an action
    persistent: percepts, a sequence, initially empty
                table, a table of actions, indexed by percept sequences, initially fully specified

    append percept to the end of percepts
    action ← LOOKUP(percepts, table)
    return action
```

The TABLE-DRIVEN-AGENT program is invoked for each new percept and returns an action each time. It retains the complete percept sequence in memory

Agent

Four basic types in order of increasing generality:

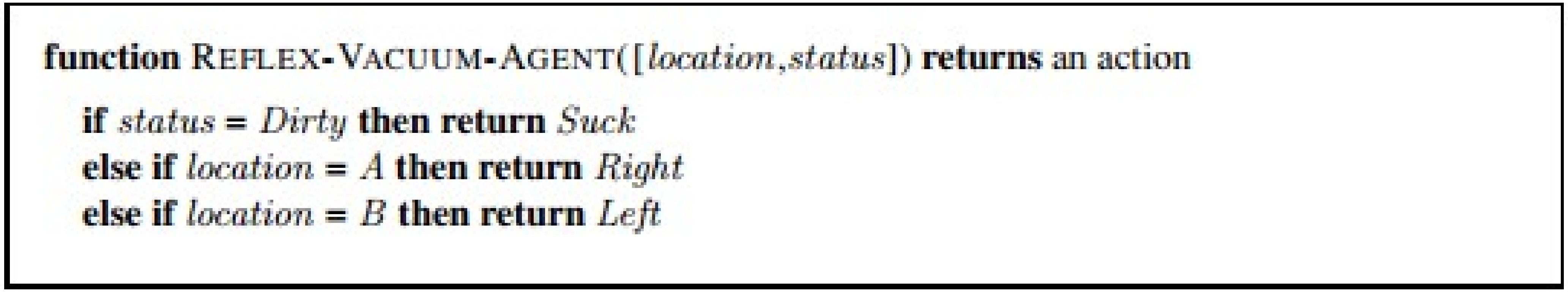
- Simple reflex agents
- Model-based reflex agents
- Goal-based agents
- Utility-based agents

Agents

Simple Reflex Agent

These agents select actions on the basis of the *current* percept, ignoring the rest of the percept history

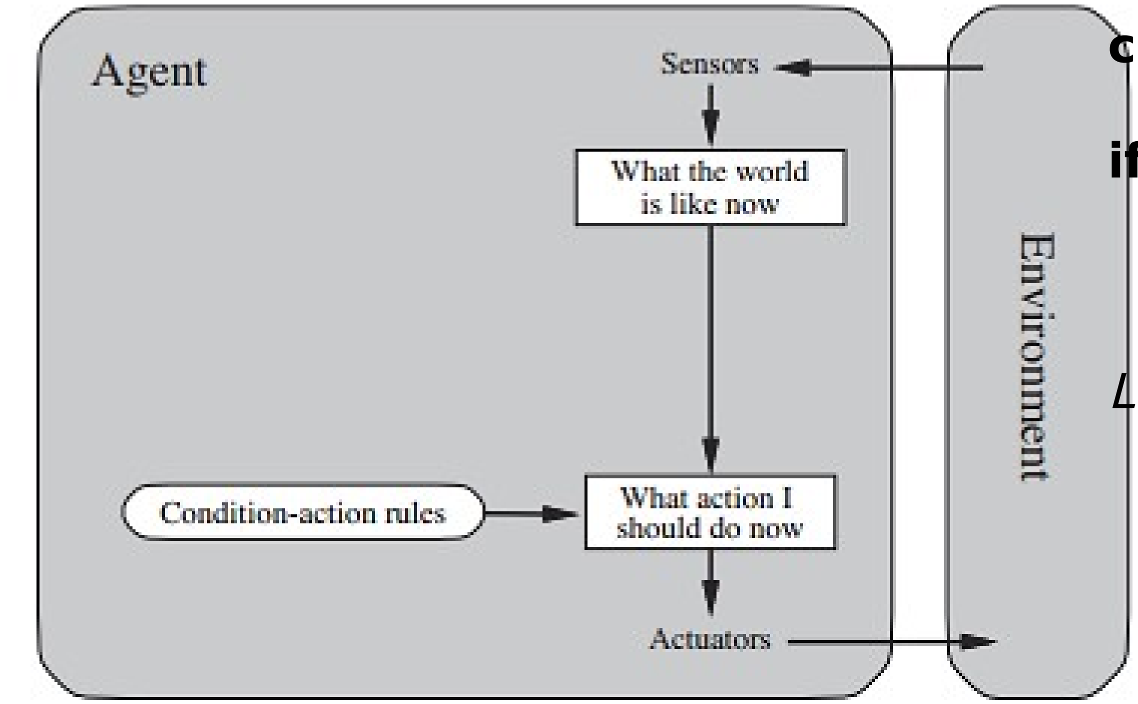
Ex: Vacuum Cleaner : its decision is based only on the current location and on whether that location contains dirt



```
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
```

Agent

Simple Reflex Agent



condition–action-rule

if *car-in-front-is-braking* **then** *initiate-braking*

Limited intelligence

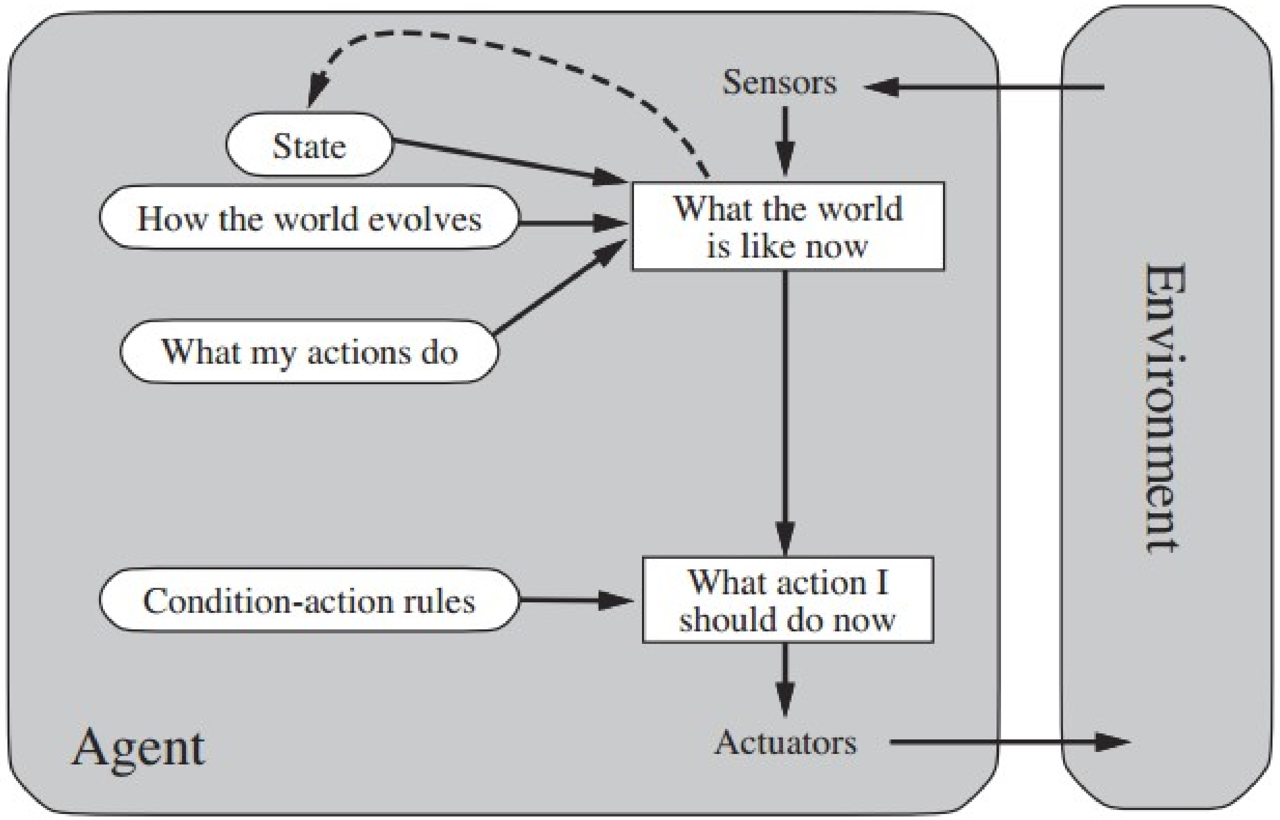
Agent

Model Based Agent

The most effective way to handle partial observability is for the agent to *keep track of the*
part of the world it can’t see now. By maintaining Internal State

how the world works:???

Model Based Agent



Agent

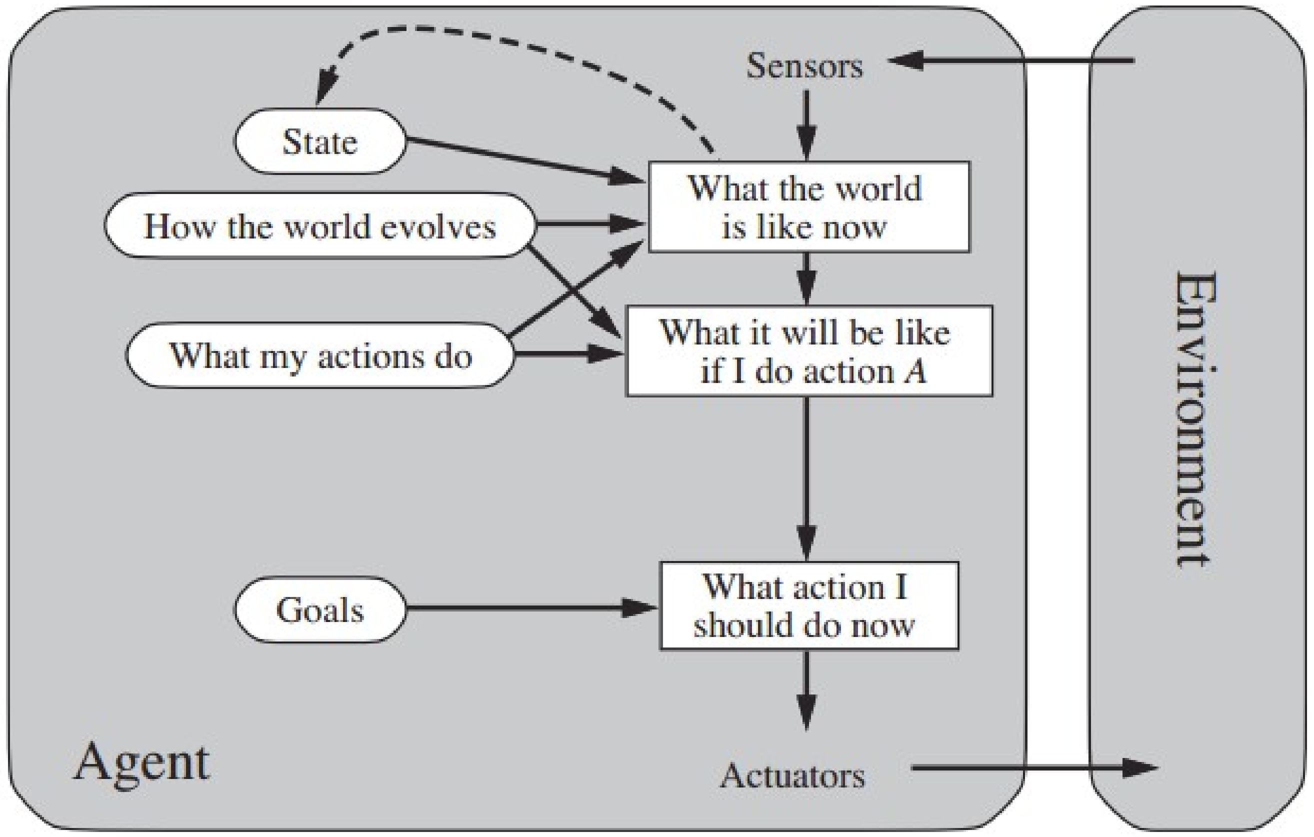
Goal Based Agent

Knowing something about the current state of the environment is not always enough to decide
what to do. For example, at a road junction, the taxi can turn left, turn right, or go straight
on.

The agent needs some sort of **goal** information that describes situations that are desirable

Agents

Goal Based Agent


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

It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

Searching and Planning is mandatory

“What will happen if I do such-and-such?” and “Will that make me happy?

Agents

Utility Based Agent

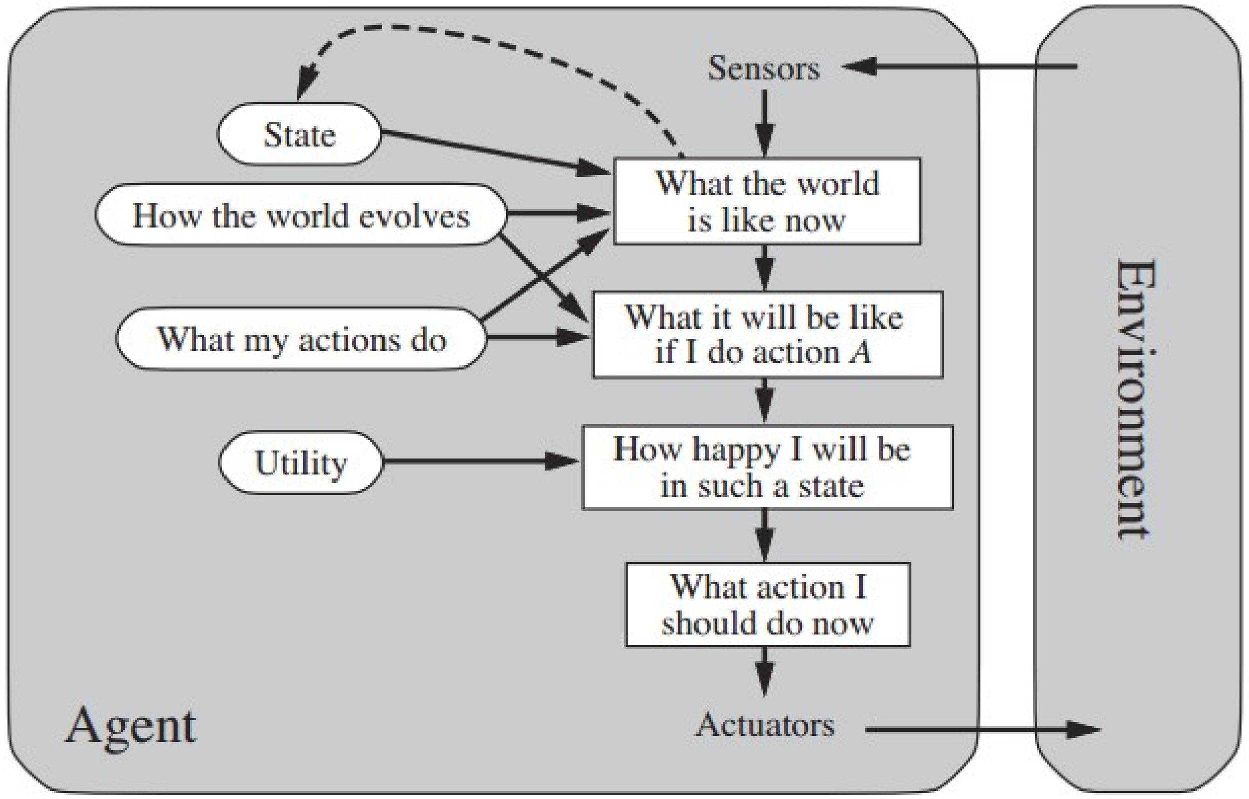
Goals alone are not enough to generate high-quality behavior in most environments. For
example, many action sequences will get the taxi to its destination (thereby achieving the
goal) but some are quicker, safer, more reliable, or cheaper than others.

Goal Achieved or not????

Happy or Unhappy

Agent

Utility Based Agents



Sensors
State
How the world evolves
What my actions do
Utility
What the world is like now
What it will be like if I do action A
How happy I will be in such a state
What action I should do now
Actuators
Agent
Environment

It uses a model of the world, along with a utility function that measures its preferences among states of the world. Then it chooses the
action that leads to the best expected utility, where expected utility is computed by averaging
over all possible outcome states, weighted by the probability of the outcome