BRAC UNIVERSITY

Inspiring Excellence

CSE422 : ARTIFICIAL INTELLIGENCE

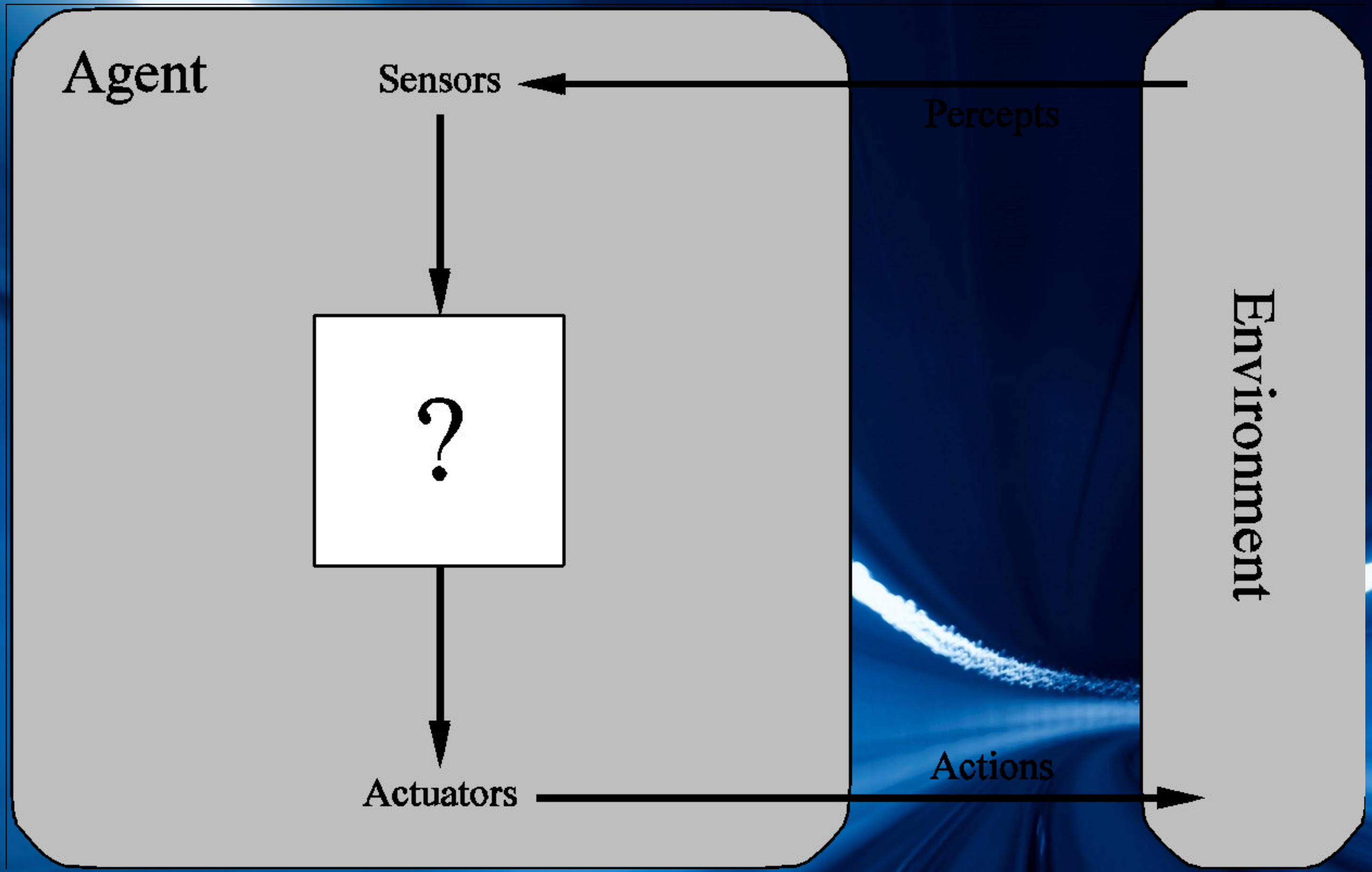
STRUCTURE OF AGENTS

BY

BENJIR ISLAM ALVEE

How do you design an intelligent agent?

- Definition: An **intelligent agent** perceives its environment via **sensors** and acts rationally upon that environment with its **actuators**.
- A discrete agent receives **percepts** one at a time, and maps this percept sequence to a sequence of discrete **actions**.
- Properties
  - Autonomous
  - Reactive to the environment
  - Pro-active (goal-directed)
  - Interacts with other agents via the environment



AGENTS

Sensors/Percepts and Actuators/Actions

- **Humans**
  - Sensors: Eyes (vision), ears (hearing), skin (touch), tongue (gustation), nose (olfaction), neuromuscular system (proprioception)
  - Percepts:
    - At the lowest level – electrical signals from these sensors
    - After preprocessing – objects in the visual field (location, textures, colors, …), auditory streams (pitch, loudness, direction), …
  - Actuators: limbs, digits, eyes, tongue, …
  - Actions: lift a finger, turn left, walk, run, carry an object, …
- **The Point:** percepts and actions need to be carefully defined, possibly at different levels of abstraction

TYPES OF AGENT

- **Table-driven agents**
  - use a percept sequence/action table in memory to find the next action. They are implemented by a (large) **lookup table**.
- **Simple reflex agents**
  - are based on **condition-action rules**, implemented with an appropriate production system. They are stateless devices which do not have memory of past world states.
- **Agents with memory(model/knowledge)**
  - have **internal state**, which is used to keep track of past states of the world.
- **Agents with goals**
  - are agents that, in addition to state information, have **goal information** that describes desirable situations. Agents of this kind take future events into consideration.
- **Utility-based agents**
  - base their decisions on **classic axiomatic utility theory** in order to act rationally.

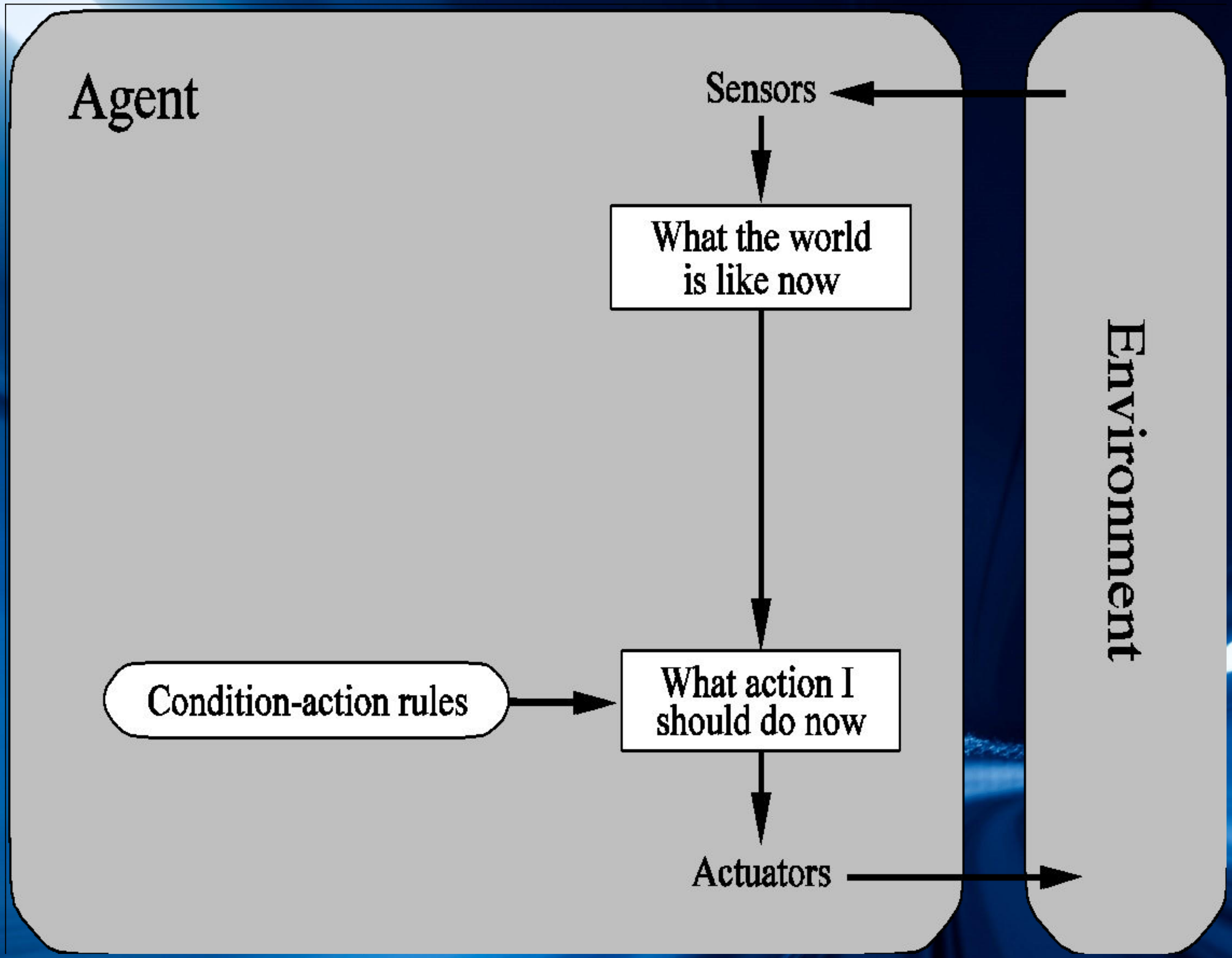
Table-driven agents

- **Table lookup** of percept-action pairs mapping from every possible perceived state to the optimal action for that state

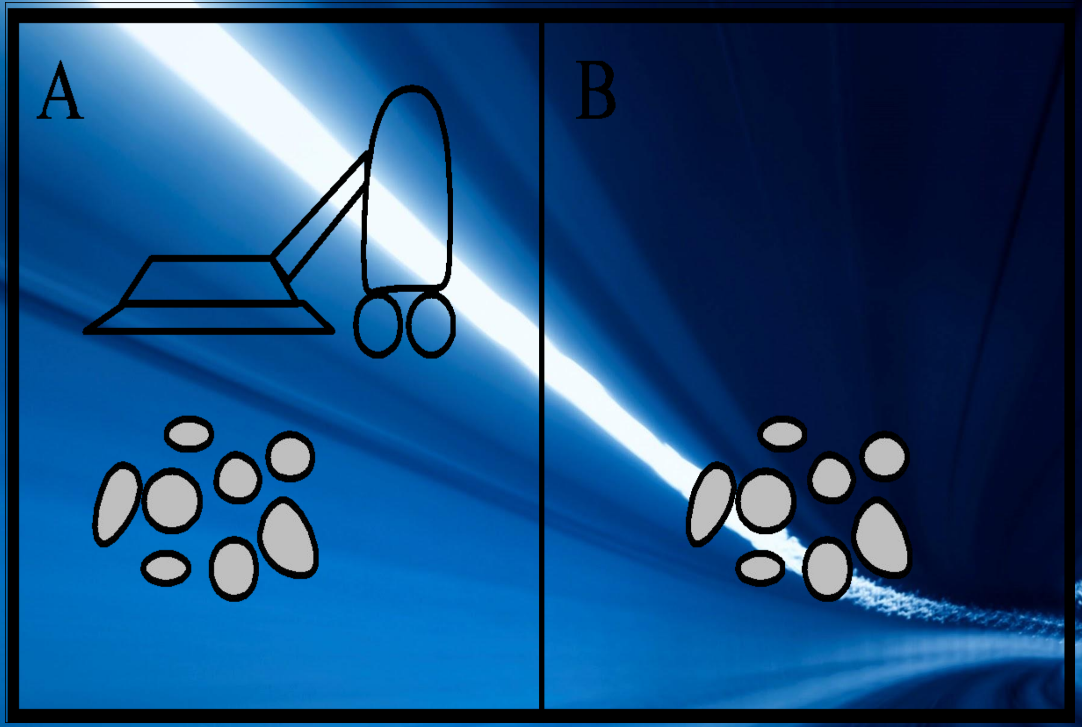
- **Problems**
  - Too big to generate and to store (Chess has about 10^{150} states, for example)
  - No agent could ever learn all the right table entries from its experience
  - No guidance about how to fill in the table entries
  - Not adaptive to changes in the environment; requires entire table to be updated if changes occur
  - Looping: Can't make actions conditional on previous actions/states

Simple reflex agents

- Immediate or spontaneous action
- Do not check past record
- Based on current state
- Based on If-Then rule
- Problems
  - Too big table to generate and to store
  - Do not work for partially observable environment
  - Do not involve multiple condition
  - No knowledge of non-perceptual parts of state
  - Not adaptive to environment change; requires collection of rules to be updated if changes occur
  - Can’t make actions conditional on previous state



Architecture of Simple reflex agent architecture



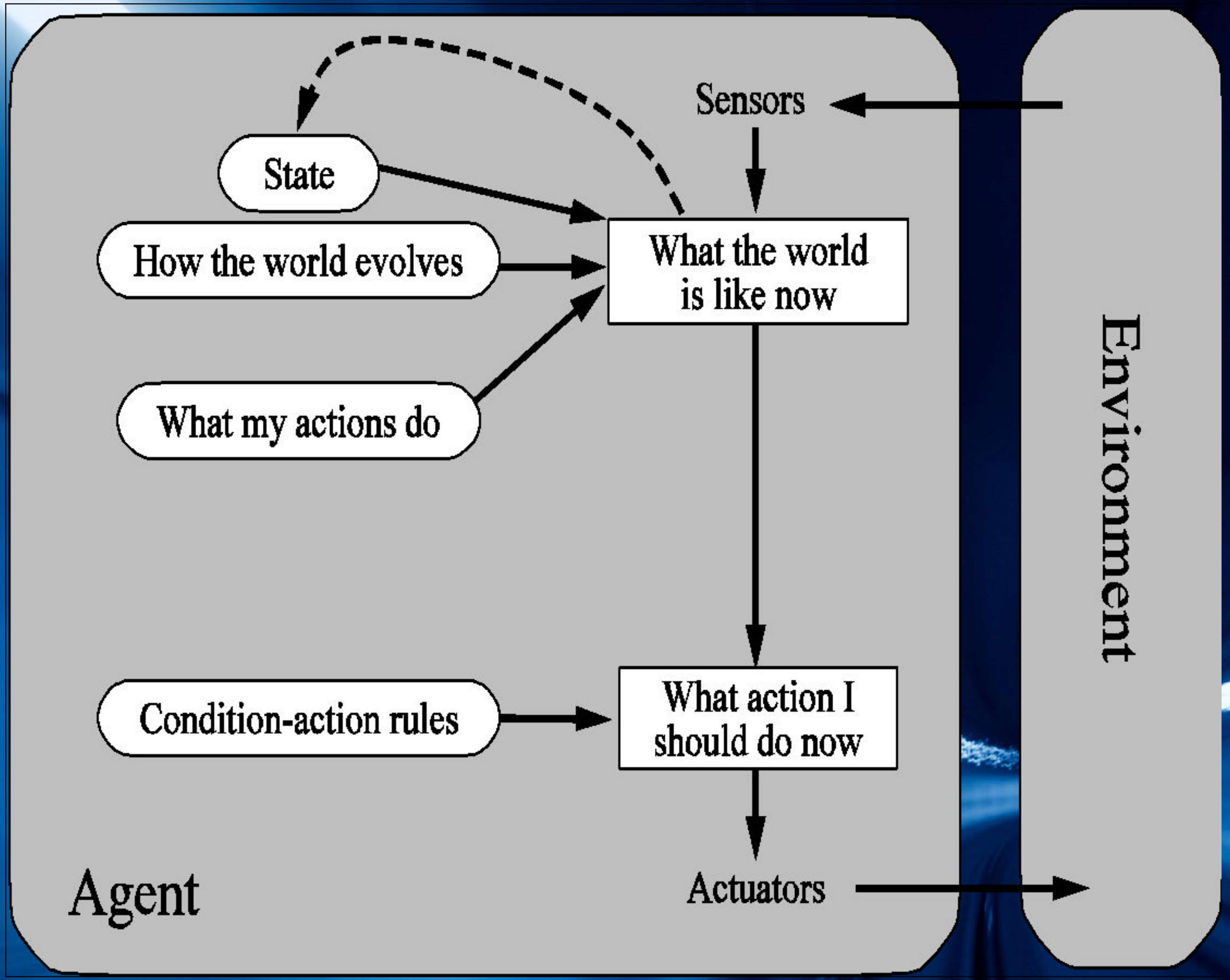
Example: Vacuum Cleaner

Simple Function

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, Dirty]	Suck

Model-based Agents

- Knowledge based
- Store percept history
- Partially observable environment
- Adaptive ability to environment change



Architecture for a model-based agent

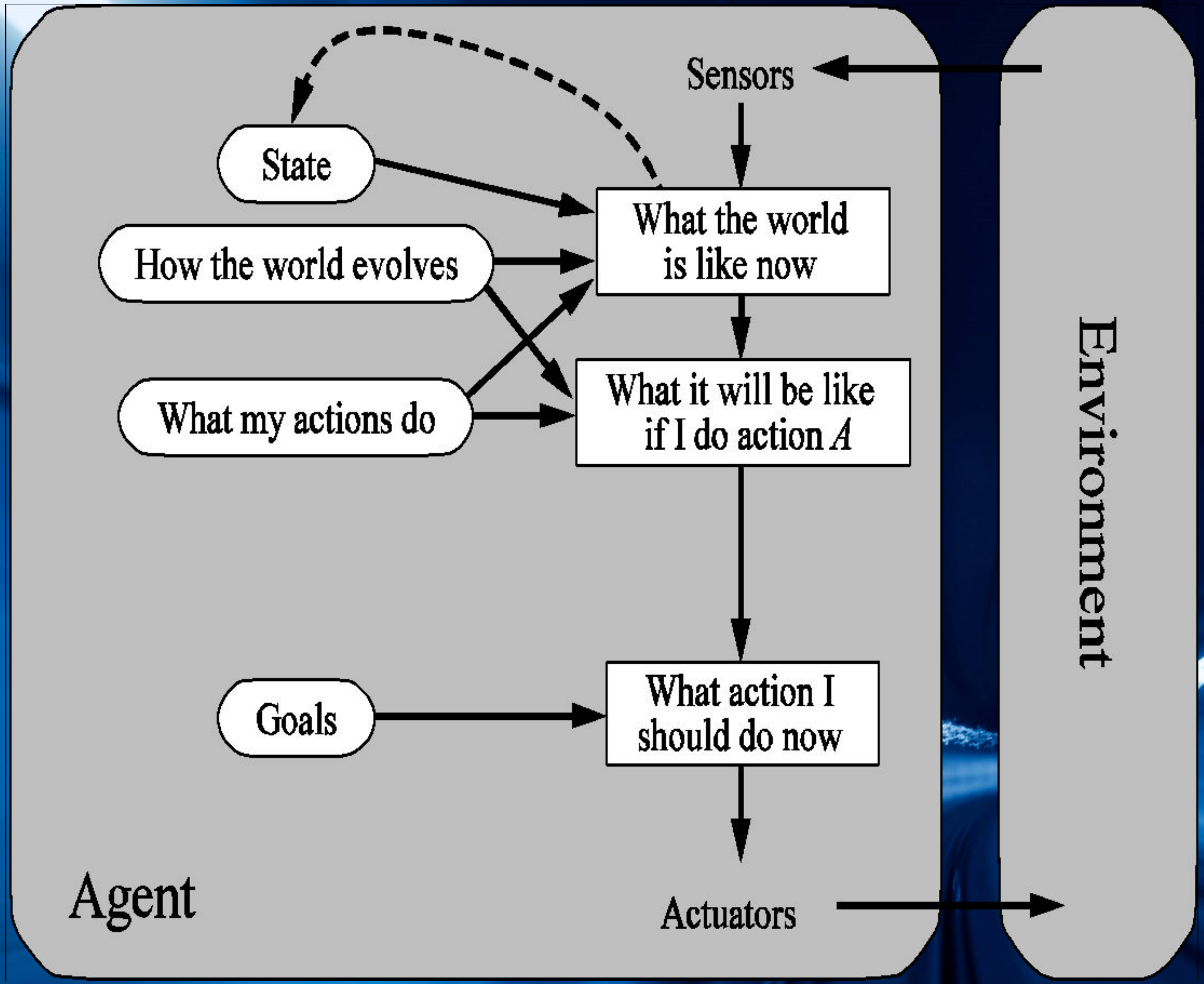
Automated taxi driving system

- **Percepts**: Video, sonar, speedometer, odometer, engine sensors, keyboard input, microphone, GPS, …
- **Actions**: Steer, accelerate, brake, horn, speak/display, …
- **Goals**: Maintain safety, reach destination, maximize profits (fuel, tire wear), obey laws, provide passenger comfort, …
- **Environment**: Urban streets, freeways, traffic, pedestrians, weather, customers, …

Goal-based agent

- Expansion of model based agent
- Choose actions so as to achieve a desired goal
- Keeping track of the current state and past state are not enough – need to add goals to decide which situations are good
- Searching and Planning
- **Deliberative** instead of **reactive**.
- May have to consider long sequences of possible actions before deciding if goal is achieved – involves consideration of the future, *"what will happen if I do...?"*

Architecture for goal-based agent

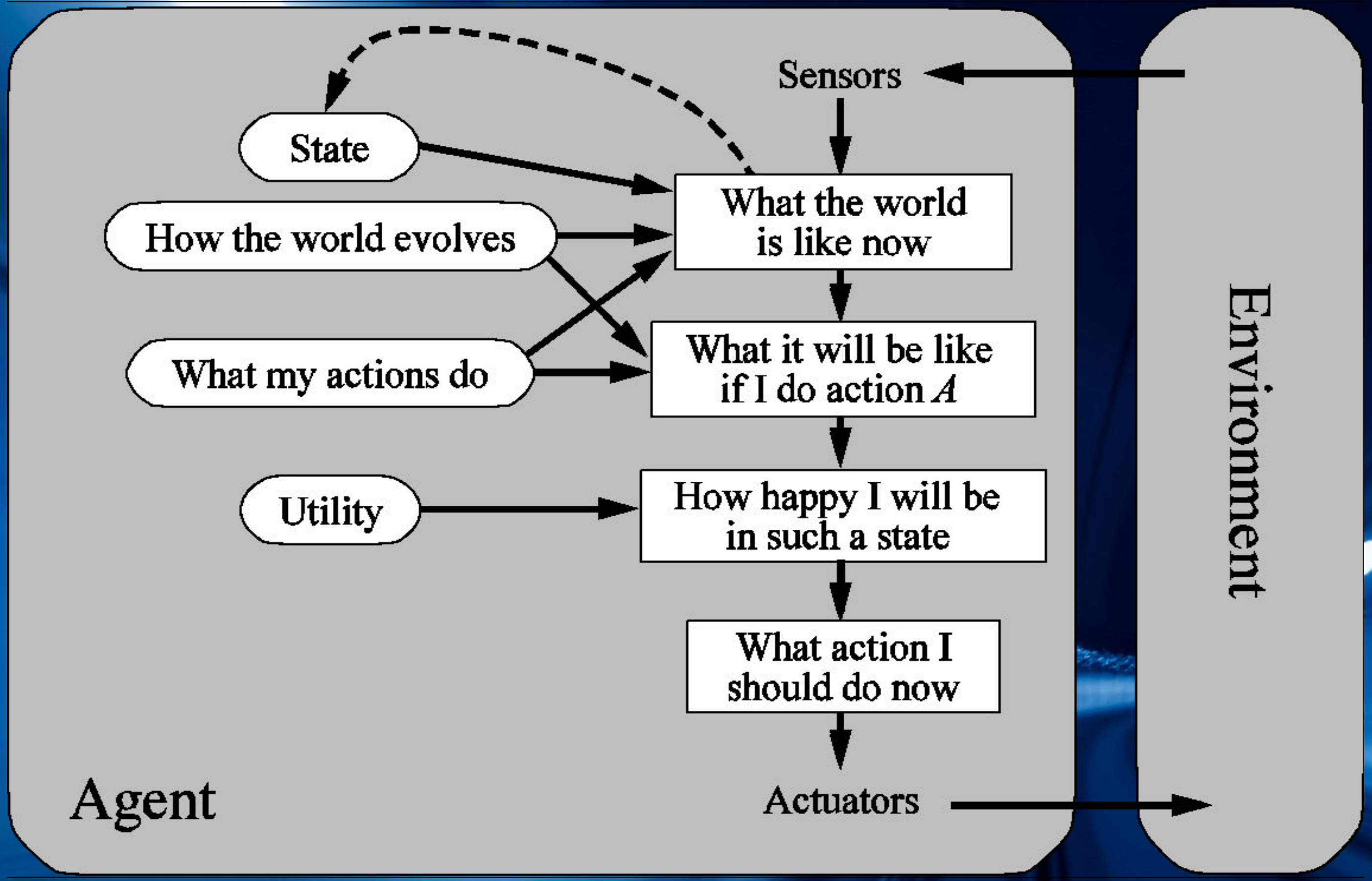


G+ Robot by Alibaba Company

Utility-based agents

- When there are multiple possible alternatives, how to decide which one is best?
- Focuses on utility or degree of happiness
- Utility function **U: State → Reals** indicating a measure of success or happiness when at a given state
- Allows decisions comparing choice between conflicting goals, and choice between likelihood of success and importance of goal (if achievement is uncertain)
- **Example: Mars Lander on the surface of mars with obstacle in its way; utility function will direct it to choose best path for best output**

Architecture for a complete utility-based agent



Thank You!