

COMP9414: Artificial Intelligence

Lecture 1b: Agents

Wayne Wobcke

e-mail: w.wobcke@unsw.edu.au

This Lecture

- Agents
- Agent Architectures and Programs
- Layered Architectures and Programs
 - ▶ Example – Delivery Robot
- Rational Agents

Three Definitions of *Agent*

- Agent as **actor**
 - ▶ Acts autonomously in the world to achieve goals
 - ▶ Rational – may have beliefs, desires and intentions
- Agent as **helper, representative**
 - ▶ Performs some task on behalf of another agent
 - ▶ Collaborates with some “user” on a (complex) task
 - ▶ May or may not be an “agent as actor”
- Agent as **catalyst 催化剂**
 - ▶ Cause of a chemical reaction

Textbooks combine the first two types, then say AI is about “agents”

Agent – Intuitive Definition

- An entity that **perceives** its environment through sensors and **acts** on its environment through effectors
- Example — human agent
 - sensors – eyes, ears, touch, etc.
 - effectors – hands, legs, etc.
- Example — robotic agent **红外线**
 - sensors – ultrasonic, **infrared** range finder, video input, etc.
 - effectors – motors, manipulators, etc.

Agents according to **Poole and Mackworth**: person, robot, dog, worm, lamp, computer program that buys and sells, corporation?

What is an Agent?

An **entity**

- **situated**: operates in a dynamically changing environment
- **reactive**: responds to changes in a timely manner
- **自主的** **autonomous**: can control its own behaviour
- **前瞻的** **proactive**: exhibits goal-oriented behaviour
- **communicating**: coordinate with other agents??

Examples: humans, dogs, ..., insects, sea creatures, ..., **thermostats**? 恒温控制器

Where do current robots sit on the scale?

Robocup Soccer



Situatedness



What does the frog **know**?

What are the implications for agent design?

Specifying Agents

- 感知** ■ **percepts**: inputs to the agent via sensors
- 行动** ■ **actions**: outputs available to the agent via effectors
- 目标** ■ **goals**: objectives or performance measure of the agent
- 环境** ■ **environment**: world in which the agent operates

Most generally, a function from percept sequences to actions

Ideally rational agent does whatever action is expected to maximize some performance measure – the agent may not **know** the performance measure (Russell and Norvig 2010)

Resource bounded agent must make “good enough” decisions based on its perceptual, computational and memory limitations (design tradeoffs)

Example Agents

Agent Type	Percepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient responses	Questions, tests, treatments	Healthy patient, minimise costs	Patient, hospital
Satellite image system	Pixels of varying intensity, colour	Print, categorisation of scene	Correct categorisation	Images from orbiting satellite
Automated taxi driver	Cameras, speedometer, GPS, sonar, microphone	Steer, accelerate, brake, talk to passenger	Safe, fast, legal, comfortable trip, maximise profits	Roads, other traffic, pedestrians, customers
Robocup robot	Camera images, laser range finder readings, sonar readings	Move motors, "kick" ball	Score goals	Playing field with ball and other robots

Based on Russell and Norvig (2010) Figure 2.5.

Representation

Example – Chess

- **States** – one way the world could be
 - ▶ e.g. 2^{43} distinct states
- **Features** – state with basic information
 - ▶ e.g. State has feature f if there is a passed pawn
- **Propositions** – state with logical structure
 - ▶ e.g. State divided into positions, pieces and relations

Agent Programs and Architectures

- Program — function implementing mapping from percept sequence to actions, using an internal representation of the percept history
- Architecture — hardware and software components, and their organization, on which agent program executes

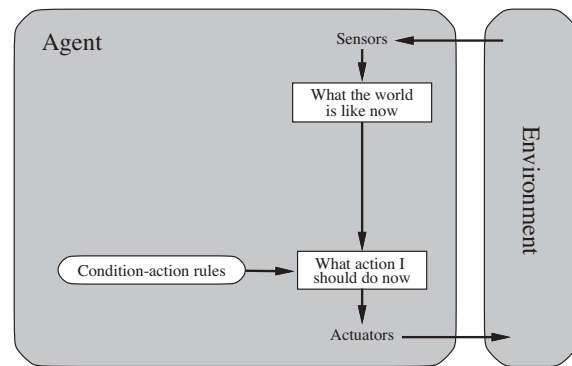
Agent = Architecture + Program

Representation

- Rich enough to express knowledge needed (to solve the problem)
- As close to the problem as possible: compact, natural, maintainable
- 有责任的 ■ **Amenable** to efficient computation
 - ▶ Able to express features of the problem that can be exploited for computational gain
 - ▶ Able to trade off accuracy and computation time and/or space
- Able to be acquired from people, data and past experiences

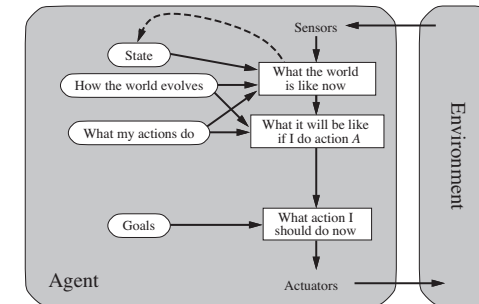
A Taxonomy of Agent Programs

Reflex (reactive) agent — applies condition-action rules to each percept



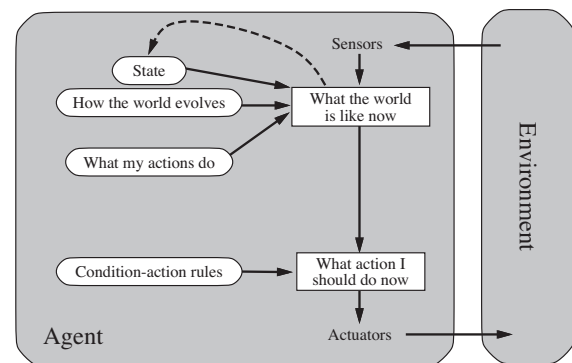
A Taxonomy of Agent Programs

Goal-based (teleological) agent — state description often not sufficient for agent to decide what to do so it needs to consider its goals (may involve searching and planning)



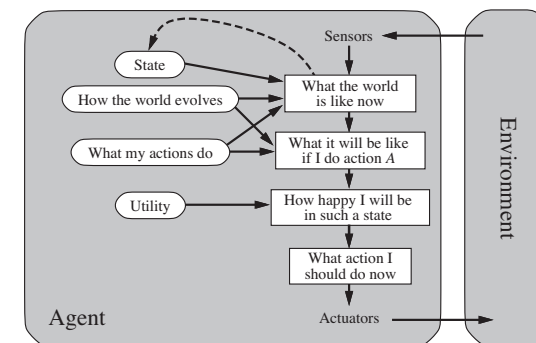
A Taxonomy of Agent Programs

Agent with internal state — keeps track of world



A Taxonomy of Agent Programs

Utility-based agent — considers preference for certain world states over others



Environment Types

Fully Observable vs Partially Observable 完全可见的 vs 部分可见的

Agent's sensors give access to complete state of environment (no internal state required)

Deterministic vs Stochastic 决定性的 vs 随机的

Next state of environment determined only by current state and agent's choice of action

Episodic vs Sequential 偶尔的、插曲的 vs 顺序的

Agent's experience divided into "episodes"; agent doesn't need to think ahead in episodic environment

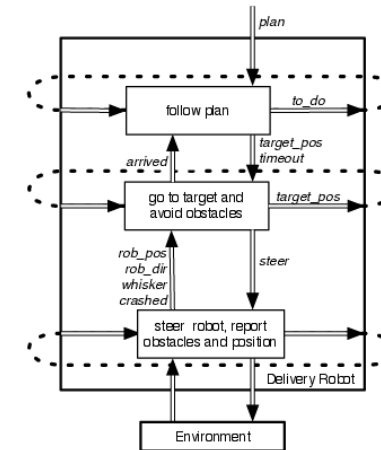
Static vs Dynamic 动态的 vs 静态的

Environment changes while agent deliberates

Discrete vs Continuous 离散的 vs 连续的

Limited number of distinct, clearly defined percepts and actions

Example – Delivery Robot

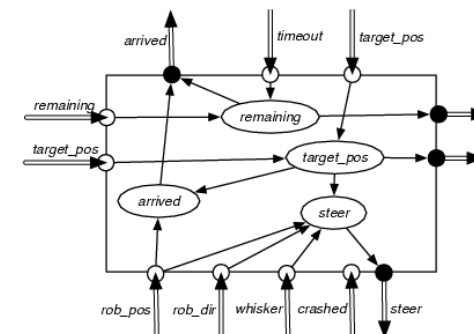


Layered Architecture

Hierarchy of controllers

- Controller gets percepts from and sends commands to the lower layer
 - ▶ Abstracts low level features into higher level (perception)
 - ▶ Translates high level commands into actuator instructions (action)
- The controllers have different representations, programs
- The controllers operate at different time scales
- A lower-level controller can override its commands

Delivery Robot – Middle Layer



Middle Layer Code

given timeout and target pos:

```

remaining := timeout
while not arrived() and remaining  $\neq$  0
  if whisker_sensor = on
    then steer := left
  else if straight ahead(rob_pos; robot_dir; target_pos)
    then steer := straight
  else if left of (rob_pos; robot_dir; target_pos)
    then steer := left
  else steer := right
  do(steer)
  remaining := remaining - 1
tell upper layer arrived()

```

Top Layer Code

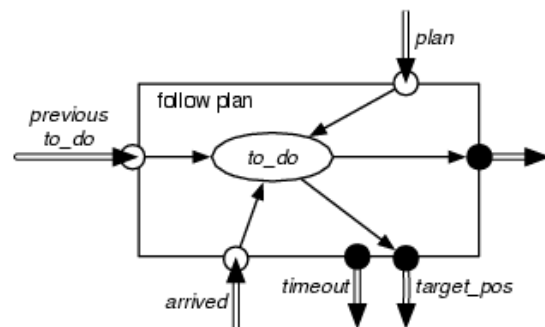
given plan:

```

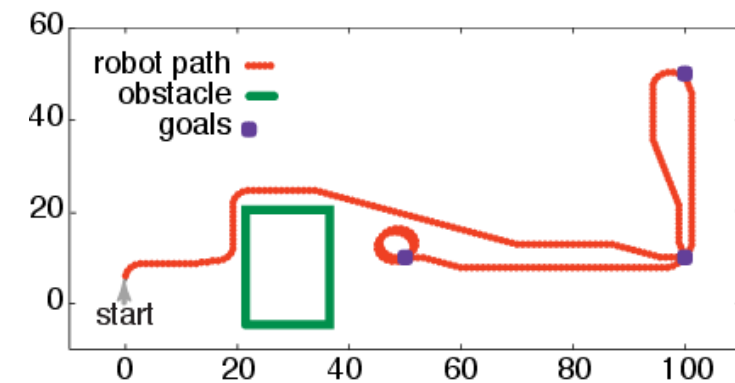
to_do := plan
timeout := 200
while not empty(to_do)
  target_pos := coordinates(first(to_do))
  do(timeout; target_pos)
  to_do := rest(to_do)

```

Delivery Robot – Top Layer



Delivery Robot – Simulation



BDI Agents

- Beliefs: Explicit representation of the world
- Desires: Preferred states of the environment
- Goals: Desires the agent has chosen to pursue (must be consistent)
- Intentions: Actions the agent has chosen and committed to
 - ▶ Pose problems for deliberation (how to fulfil them)
 - ▶ Constrain further choices (must be compatible)
 - ▶ Control conduct (lead to future action)

All defined **functionally** – no X factor

“Intentions, Plans, and Practical Reason” (Bratman 1987)

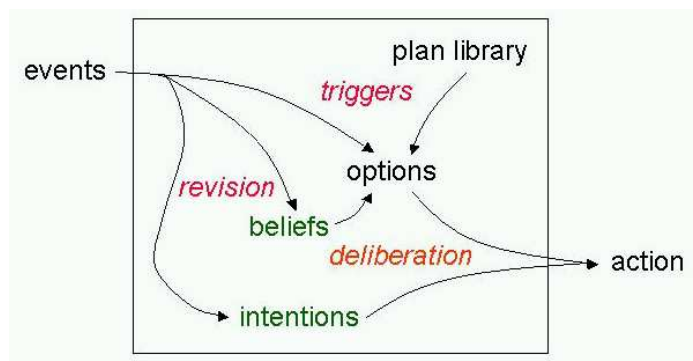
PRS (Procedural Reasoning System)

Abstract PRS Interpreter:

```

initialize-state()
do
  options := option-generator(event-queue, B, G, I)
  selected-options := deliberate(options, B, G, I) 考量、商议
  update-intentions(selected-options, I)
  execute(I)
  get-new-external-events()
  drop-successful-attitudes(B, G, I)
  drop-impossible-attitudes(B, G, I)
until quit
  
```

BDI Agent Interpreter



PRS (Procedural Reasoning System)

- Useful in dynamic environments where
 - ▶ Reasonable plans can be formed in advance
 - ▶ Agent needs continuity of commitment
 - ▶ Agent needs to respond rapidly to situation
 - ▶ Agent's computational resources are limited
 - ▶ Agent can keep up with changes in the world

Summary

听写、指示

- Assumptions made about environment **dictate** nature of agent
 - ▶ Capabilities those needed to survive in the environment
 - ▶ Need not be “over-engineered” to handle more complexity
 - ▶ Agent + Environment can be thought of as a coupled system
- Specific architectures constrain agent’s computational power and limit behaviour: more efficient than general architectures