COMP9414: Artificial Intelligence

Intelligent Agents

Wayne Wobcke

Room J17-433

wobcke@cse.unsw.edu.au

Based on slides by Maurice Pagnucco

Overview

- Rational Agents
- Taxonomy of Agent Programs
- Environments
- Coupling Agents to Environments
- BDI Agents
- Reference: Stuart J. Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Second Edition, Pearson Education, 2003. (Chapter 2)

What is an Intelligent Agent?

- Agent — 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

  sensors – ultrasonic, infrared range finder, video input, etc.
  effectors – motors, manipulators, etc.

Rational Agents

- We would like to design and build rational agents
- Rational agent – an agent that does the right thing
- But what is right?
- Initial idea: “right thing” to do is that which makes the agent most “successful”

Rational Agents

- Rationality depends on:
  - The performance measure that defines degrees of success
  - Everything agent has perceived so far (percept sequence)
  - What agent knows about its environment
  - Actions agent can perform
- Ideal Rational Agent:

  For each possible percept sequence, an ideal rational agent should do whatever is expected to maximise its performance measure, on the basis of the evidence provided by the percept sequence and whatever built-in knowledge the agent has

Mappings

- Therefore, agent's behaviour depends only on percept sequence
- Mapping – describes agent via a table: entries correspond to action(s) taken in response to each percept sequence
- In principle (but not always in practice) it is easy to determine
- Ideal mapping – which action(s) agent ought to take in response to given percept sequence
- A mapping can be specified by a table or a program

Autonomy

- An agent is autonomous to the degree that its behaviour is determined by its experience/perception
- Need to provide agent with initial knowledge plus ability to learn

Agent Programs and Architectures

- Agent program — function implementing mapping from percept sequence to actions
- Architecture — computing device on which agent program will run

Agent = Architecture + Program

e.g. can have robotic agents, software agents (softbots, infobots), etc.

Agents

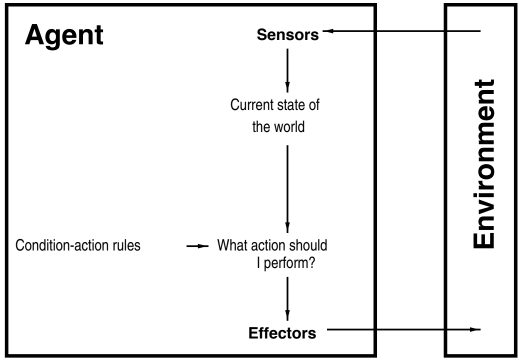
Agent Type	Percepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient responses	Questions, tests, treatments	Healthy patient, minimise costs	Patient, hospital
Satellite image system	Pixels of varying intensity, colour	Print categorisation of scene	Correct categorisation	Images 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 (1995) Figure 2.3.

A Taxonomy of Agent Programs

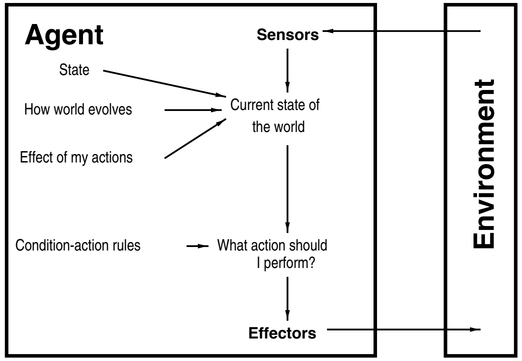
Modelled after (Russell and Norvig, 1995)

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



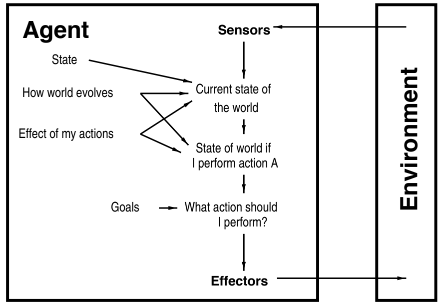
A Taxonomy of Agent Programs

Agent with internal state — keeps track of world



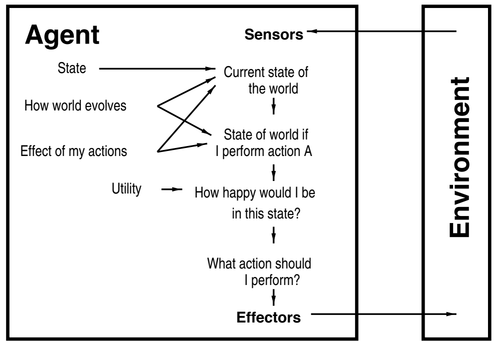
A Taxonomy of Agent Programs

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

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



Environments

Accessible vs. Inaccessible

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

Deterministic vs. Non-deterministic

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

Episodic vs. Non-episodic

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

Static vs. Dynamic

environment changes while agent deliberates

Discrete vs. Continuous

limited number of distinct, clearly defined percepts and actions

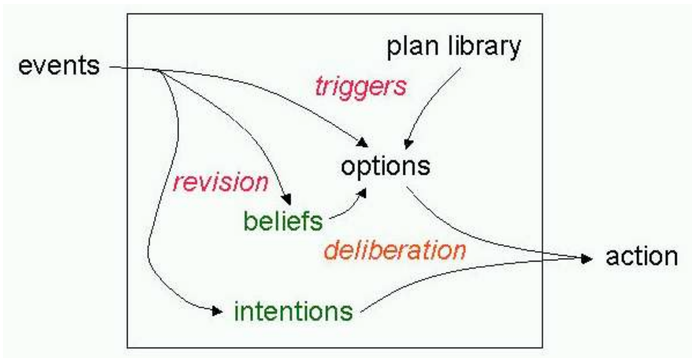
Agents

- Why are all these considerations important?
- Assumptions made about environment dictate nature of agent
- Need only design agent complex enough to deal with its environment
- Determine how agent will interact (couple) with environment
- Specific architectures constrain agent's computational power and limits behaviour: aim to be more efficient than general architectures

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)

BDI Agent Interpreter



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);
    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
```


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

Conclusion

- The term "agents" has become very widespread in recent literature yet the meaning of the term is very unclear (arguably because it is used in vague terms and it means different things to different people!)
- We have tried to give a definition which is broad yet encompasses much of the work we are trying to do
- Keep in mind that we are primarily concerned with techniques that can be used to build components of an agent not the entire agent itself
- Is the technique's use limited to only certain of the environments that we have discussed? Is it widely applicable?