Agents

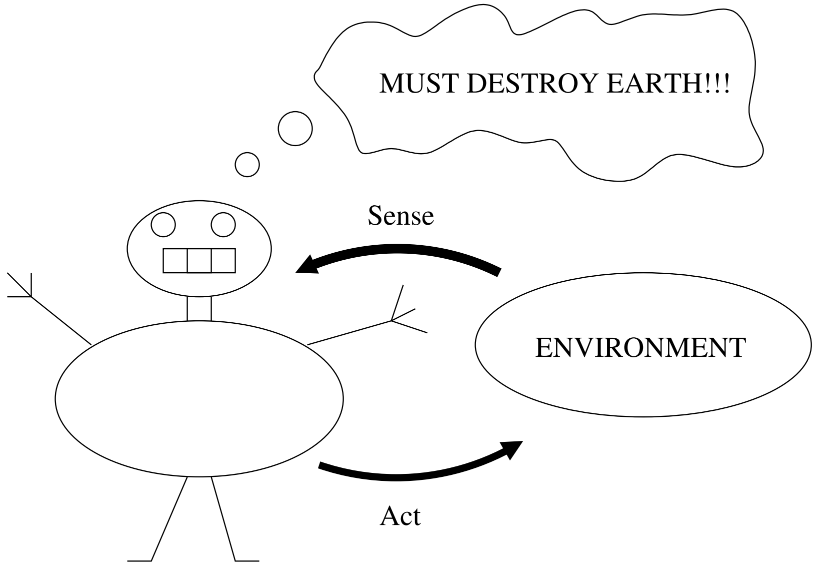
We now look at a simple unifying concept for the construction of AI systems: the idea of an *agent*. **Aims:**

- to introduce agents as a way of speaking about a wide range of AI systems;
- to look at some ways in which agents might be structured;
- to connect these structures to standard fields of study within AI as a subject;
- to look briefly at ways in which an agent's *environment* is significant.

Reading: Russell and Norvig, chapter 2.

Agents

There are many different definitions for the term *agent* within AI. (Be aware of this when reading beyond the course textbook.)



We will use the following simple definition: *an agent is any device that can sense and act upon its environment*.

Agents

This definition can be very widely applied: to humans, robots, pieces of software, and so on. It is only one of many.

Questions:

- How can we judge an agent's performance?
- How can we begin successfully to design an agent?
- How can an agent's environment affect its design?

Recall that we are interested in devices that *act rationally*, where 'rational' means doing the *correct thing* under *given circumstances*.

Measuring performance

Clearly any *performance measure* we apply will need to be domain-dependent, so we will have to design one appropriate for a given problem.

Example: for a chess playing agent, we might use its rating.

Example: for a mail-filtering agent, we might devise a measure of how well it blocks spam, but allows interesting email to be read.

Example: for a car cleaning robot, we might want maximum removal of dirt in minimum time. Being more sophisticated, perhaps we don't want the car to get damaged, or to use too much water or energy, and we might want the robot to have spare time at the weekend to write its novel...

So: the choice of a performance measure can be tricky.

Measuring performance

Two further points:

- In general, we will be interested in success *over the long term*. For example, we might not want to favour a car-cleaner that's extremely fast in the first hour and then sits around reading, over one that works consistently.
- We are generally interested in *expected* performance because usually agents are not *omniscient*—they don't *infallibly* know the outcome of their actions.

It is *rational* for you to enter this lecture theatre even if the roof falls in today.

Measuring performance

So rational behaviour requires us to know:

- A well-defined measure of performance.
- What our agent has already perceived. The *percept sequence*.
- What our agent knows about the environment it lives in.
- What actions our agent is capable of performing.

An *ideal rational agent* acts as follows: for any percept sequence it acts so as to expect to maximise performance, given what it knows about the world via the percept sequence and its own knowledge.

So: an agent capable of detecting and protecting itself from a falling roof might be more *successful* than you, but *not* more *rational*.

Environments

Some common attributes of an environment have a considerable influence on agent design.

- **Accessible/inaccessible:** do percepts tell you *everything* you need to know about the world?
- **Deterministic/non-deterministic:** does the future depend predictably on the present and your actions?
- **Episodic/non-episodic** is the agent run in independent episodes.
- **Static/dynamic:** can the world change while the agent is deciding what to do?
- **Discrete/continuous:** an environment is discrete if the sets of allowable percepts and actions are finite.
- **Single-agent/multi-agent:** is the agent acting individually or in the presence of other agents. In the latter case is the situation *competitive* or *cooperative*, and is *communication* required?

Basic structures for intelligent agents

Example: email spam filter.

Percepts: the textual content of individual email messages. (A more sophisticated program might also take images or other attachments as percepts.)

Actions: send to the inbox, delete, or ask for advice.

Goals: remove spam while allowing valid email to be read.

Environment: an email program.

Basic structures for intelligent agents

Example: aircraft pilot.

Percepts: sensor information regarding height, speed, engines *etc*, audio and video inputs, and so on.

Actions: manipulation of the aircraft's controls. Also, perhaps talking to the passengers *etc*.

Goals: get to the current destination as quickly as possible with minimal use of fuel, without crashing *etc*.

Environment: aircraft cabin.

Mapping percepts to actions

In principle, we could simply enumerate what actions an ideal agent should take given *any* percept sequence.

- We could produce a table and store it.
- Or we could produce a program capable of reproducing such a table.

For example: a square-root agent.

This seems simplistic and potentially trivial, *but* the basic idea can be expanded considerably.

Programming agents

A basic agent program is as follows:

```
action agent(percept)
{
   static memory;       // the agent's memory.

   memory = update_memory(memory,percept);
   next_action = choose_action(memory);
   memory = update_memory(memory,next_action);

   return next_action;
}
```

It is up to the agent how long a list of past percepts it stores, and the measure of performance is not known to the agent.

Programming agents

The simplest approach would be to use a table to map percept sequences to actions, but this can quickly be rejected.

- The table will be *huge* for any problem of interest. About 35^{100} entries for a chess player.
- We don't usually know how to fill the table.
- Even if we allow table entries to be *learned* it will take too long.
- The system would have no *autonomy*.

We can overcome these problems by allowing agents to *reason*.

Autonomy

If an agent's behaviour depends in some manner on its *own experience of the world* via its percept sequence, we say it is *autonomous*.

- An agent using only built-in knowledge would seem not to be successful at AI in any meaningful sense: its behaviour is predefined by its designer.
- On the other hand *some* built-in knowledge seems essential, even to humans.

Not all actual animals are entirely autonomous. **For example:** dung beetles.

Reflex agents

We can’t base our example spam filter on a table: there are too many character sequences to consider.

But we might try *extracting pertinent information* and using *rules* based on this.

Condition-action rules:

if a certain *state* is observed **then** perform some action

Example:

if message contains ‘gambling’ and ‘online’ **then** delete

Reflex agents

```
action agent(percept)
{
   static rules;          // A collection of
                          // condition-action rules.

   state = interpret(percept);
   rule = match(state,rules);
   next_action = find_action(rule);

   return next_action;
}
```

Keeping track of the environment

Some points immediately present themselves regarding reflex agents:

- we can't always decide what to do based on the current percept;
- however storing *all* past percepts might be undesirable (for example requiring too much memory) or just unnecessary;
- reflex agents don't maintain a description of the state of their environment;
- however this seems necessary for any meaningful AI. (Consider automating the task of driving.)

This is all the more important as usually percepts don't tell you everything about the state.

Keeping track of the environment

An agent should maintain:

- a description of the current state of its environment;
- knowledge of how the environment changes independently of the agent;
- knowledge of how the agent's actions affect its environment.

This requires us to do *knowledge representation* and *reasoning*.

Keeping track of the environment

```
action agent (percept)
{
   static state;
   static rules;

   state = update(state,percept);
   rule = match(state,rules);
   next_action = find_action(rule);
   state = update(state,action);

   return next_action;
}
```

Goal-based agents

Sometimes, choosing a rational course of action depends on your *goal*.

- We need to consider *goal-based agents*.
- As agents include knowledge of how their actions affect the environment, they have a basis for choosing actions to achieve goals.
- To obtain a *sequence* of actions we need to be able to *search* and to *plan*.

This is *fundamentally different* from a reflex agent. For example, by changing the goal you can change the entire behaviour.

Utility-based agents

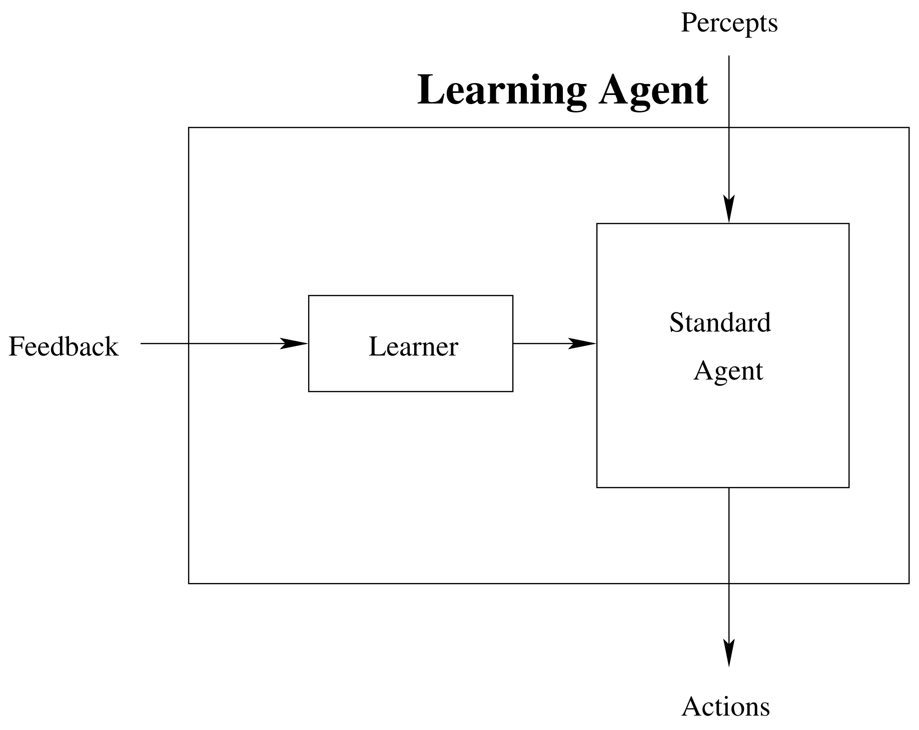
Introducing goals is still not the end of the story.

There may be many sequences of actions that lead to a given goal, and some may be preferable to others.

A *utility function* maps a state to a number representing the desirability of that state.

- We can trade-off *conflicting goals*, for example speed and safety.
- If an agent has several goals and is not certain of achieving any of them, then it can trade-off likelihood of reaching a goal against the desirability of getting there.

Learning agents



Here, the *learner* needs some form of *feedback* on the agent's performance. This can come in several different forms. In general, we also need a means of *generating new behaviour* in order to find out about the world.