Dipartimento di elettronica e informazione
Politecnico di Milano

Artificial Intelligence 2010-11

2. Rational Agents

2.1 What is a rational agent

An *agent* is any physical system that is able to *act* in an *environment*. Human beings and other animal organisms are agents; also certain artificial systems can be considered as agents, possibly in a metaphorical sense.

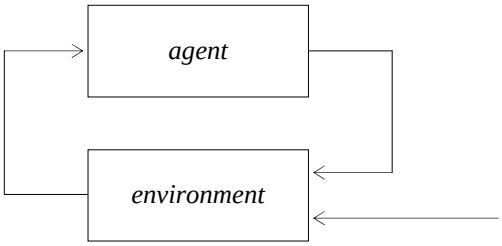
Rational agent are agents that act on the basis or *reasons*. *Reasoning* is the process by which reasons are taken into account: the term comes from the Latin verb *reri*, which means both *to judge* and *to compute*. Reasoning to decide whether something is the case is called *theoretical reasoning*, and reasoning to choose the best course of action is called *practical reasoning*. Even if we are mainly interested in rational action, it must be noted that practical reasoning relies heavily on theoretical reasoning, because the choice of how to act depends also on what an agent believes about the environment in which actions are to be carried out.

Economical theories of rationality assume that rational agents choose their actions on the basis of a predefined set of *preferences*. In some cases preferences are assumed to be totally ordered by a numerical function whose value has to be maximized (and is then called *utility* or *payoff*) or minimized (and is then called *cost*). These theories provide a very narrow view of human rationality; in particular, even if one accepts that acting rationally consists of acting on the basis of a set of preferences, the most difficult problem appears to be that of *establishing a sensible set of preferences*, rather than choosing what to do on the basis of predefined preferences. In any case, the concept of rationality that is adopted in AI is close to the economical perspective.

2.2 Analysing rational agents

At a very high level of abstraction, we can view an agent as a "black box" system acting on an environment (Scheme 1). In turn, the agent's environment can be defined as the smallest part of the universe that may affect, and be affected by, the agent's actions.

Scheme 1:



Two points are stressed in Scheme 1:

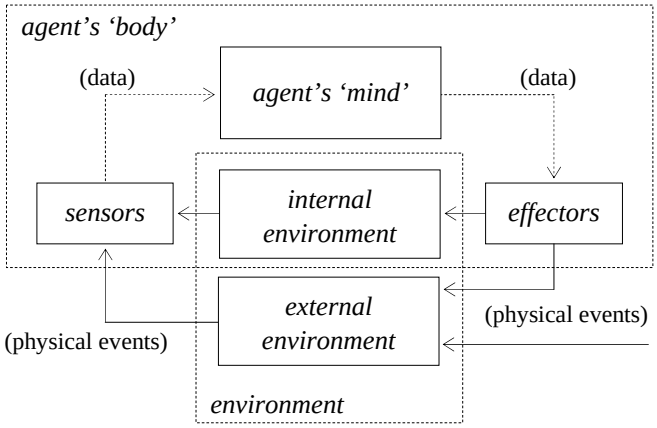
- An agent and an environment come as a pair. Given that (i) *an agent is a system that acts*, and that (ii) *acting involves interacting with some external system*,[1] there can be no agent without an environment.

[1] The are some exceptions, that is, actions that can be performed without interacting with an external environment. Consider for example the purely mental action of tacitly adding 5 and 3. But a system able to perform only mental actions, without ever interacting with an external environment, could not be considered as an agent proper.

- By acting, agents tries to *control* their environments. However, the temporal evolution of an environment is partially unpredictable, also due to the fact that it receives inputs that are not determined by the agent.

The term "agent" conceals an ambiguity that becomes clear if one thinks of a biological agent (i.e., an animal organism) or of a robot. What do we mean exactly by *an agent*, just its 'mind' or also its 'body' (inclusive of the mind)? This point is made clearer by Scheme 2:

Scheme 2:



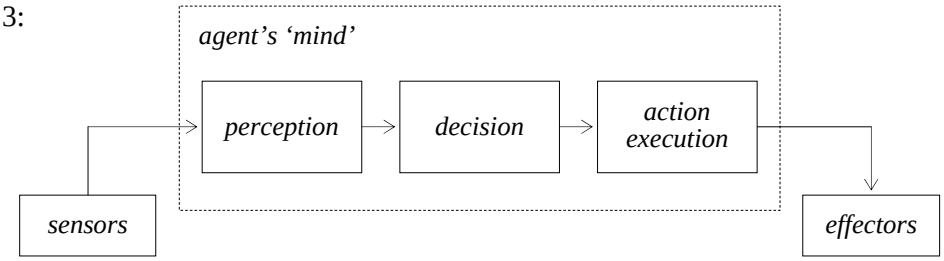
The agent, as a whole, includes *sensors* and *effectors* (or *actuators*), that is, the interface of the agent with its environment. Both sensors and effectors are *tranducers*:

- sensors transform *physical events* into *data*, and
- effectors transform *data* into *physical events*.

Often a part of the environment, called *internal environment*, is internal to the agent's 'body.' For example, a robot's battery is part of the environment, but also part of the robot's physical system. Sensors and effectors may interact with the internal environment: for example, a robot may have sensors to monitor the charge level of its batteries.

The agent's 'mind,' which receives data from the sensors and sends data to the effectors, may be broken down into three main modules as shown in Scheme 3.

Scheme 3:



The function of *perception* is to transform the data coming from the sensors into partial representations of the current state of the environment, that can be used to decide what to do. In turn, the output of the *decision* module consists of *representations of actions*, which are translated by the *action execution* module into data that drive the effectors.

Perception (with real sensors) and action execution (with real effectors) are very relevant in robotics, but less so in applications where agents are software applications without a mechanical 'body,' like for example software agents acting in the Web. In this course, in which robots are not a main concern, we shall concentrate on the decision module.

2.3 A classification of agents based on the decision module

The decision module may have different degrees of complexity.

Simple reflex agents

The simplest type of agent can be modelled as follows:

- Independently of whether the environment is modelled in discrete or continuous time, the *agent's time* is determined by a discrete clock and can be modelled as a sequence T of *time instants*.
- At every time instant t, the agent perceives the environment as p_t (which is a partial representation of the current state of the environment).
- The agent's decision module implements a function D that directly maps representation p_t onto a representation $a_t = D(p_t)$ of the next action to be executed.
- Function D is designed so that the agent achieves high (possibly maximum) utility.

An agent of this type is called a *simple reflex agent*.

(Example 1: A robot collecting golden nuggets)

From the point of view of AI, the most interesting case is when function D is allowed to be *time variant*, that is, when

$$a_t = D_t(p_t).$$

In this case one can try to solve the problem of *learning*, that is, of improving function D_t with time, in such a way that for increasing values of t, function D_t allows the agent to achieve higher utility. To this purpose, methods of *reinforcement learning* may be used (see the course on *Soft Computing*). For the representation of function D_t, the learning process may exploit simple *perception-action tables*, or *classifier systems*, or *neural networks*.

Model-based reflex agents

What happens when relevant aspects of the environment cannot be perceived by the agent? This may be because a relevant event happened in the past, or because a relevant state of affairs is beyond the reach of agent's sensors. In such cases, an agent may exploit a *model* of its environment.

(Example 2: The desert ant)

This type of agent needs a more complex decision function, and also a model-management function:

$$a_t = D(p_t,m_t),$$
$$m_{t+1} = M(p_t,m_t),$$

where m_t is a model of the relevant parts of the environment that cannot be directly perceived. A *map of a territory* is a good example.

Goal-driven agents

So far we have implicitly assumed that an agent has a single *goal*, which is 'hardwired' in its decision function. On the contrary, *goal-directed* agents rely on explicit representations of their goals. Typically, a goal-directed agent has one or more *final goals*, which it strives to achieve. In trying to do so, an agent may have to generate new intermediate goals (also called *subgoals*). The whole mechanism of goal setting is designed so that, in general, high (possibly maximum) utility can be achieved.

$$a_t = D(p_t,m_t,g_t),$$
$$m_{t+1} = M(p_t,m_t,g_t),$$
$$g_{t+1} = G(p_t,m_t,g_t).$$

To deal with subgoals, function *G* may manage a *goal stack*:

- initially, a final goal g_f is pushed onto the stack;
- at every instant *t*, the agent's current goal g_t is the goal on top of the stack;
- when reaching goal *g* makes it necessary to previously reach goal *g'*, goal *g'* is pushed on top of the stack;
- when a goal is reached, it is popped from the stack.

In general, however, more complex goal management strategies are required, and this will be the main topic of this course.

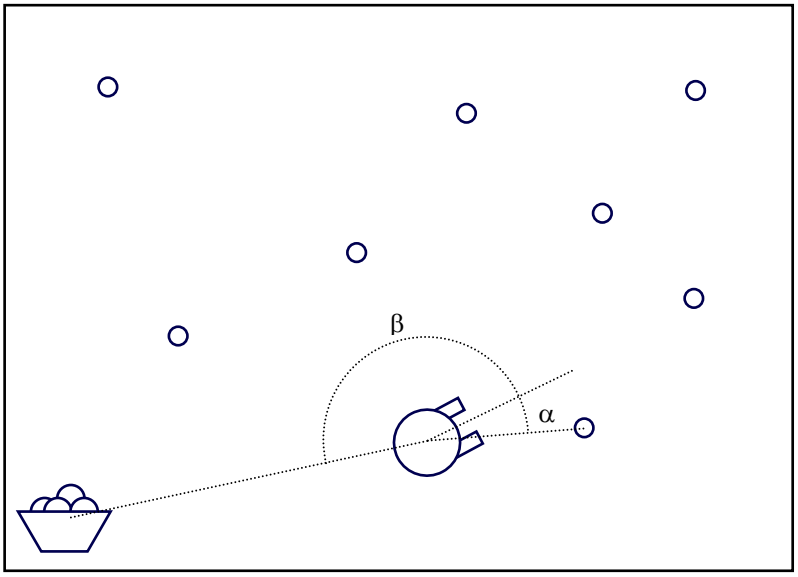
Utility-driven agents

In the three types of agents we have analysed so far, we assume that the decision function, the model-management function, and the goal-management function are designed so that the agent will achieve high (or even maximum) utility.[2] Note, however, that the utility function is not assumed to be explicitly represented inside the decision module.

Agents, however, may include explicit representations of utility functions in their models of the environment. This enables an agent, for example, to chose its own final goals so that utility is maximised. Agents able to do so are called *utility-driven agents*.

[2] In the case of goal-driven agents, also the choice of the final goal or goals affects the agent's utility.

Example 1: A robot collecting golden nuggets



Perceptions:

- nuggetThere(α): a nugget is perceived at angle α
- basketThere(β): the basket is perceived at angle β
- nuggetHere( ): the robot is at a nugget
- basketHere( ): the robot is at the basket
- holding( ): the robot's hand is holding an object

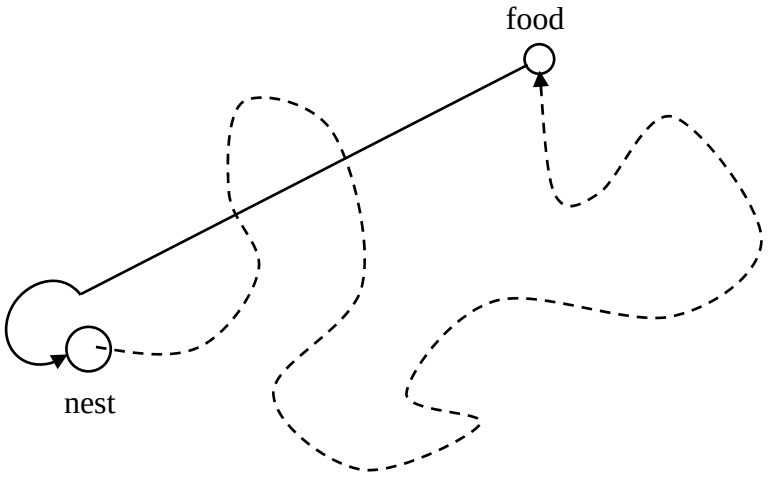
Actions:

- turn(α): turn an angle α
- goAhead( ): move ahead
- grasp( ): grasp an object
- ungrasp( ): ungrasp
- explore: move randomly

Decision rules (with associated priorities: 1 = high, 5 = low):

1: holding( ) **and** basketHere( ) → ungrasp( )
2: holding( ) **and** basketThere(0) → goAhead( )
2: holding( ) **and** basketTheret(β) **and** $\beta \neq 0$ → turn(β)
3: nuggetHere( ) → grasp( )
4: nuggetThere(0) → goAhead( )
4: nuggetThere(α) **and** $\alpha \neq 0$ → turn(α)
5: **true** → explore( )

Example 2: The desert ant



The desert ant (*Cataglyphis bicolor*, `http://en.wikipedia.org/wiki/Sahara_Desert_ant`) explores its territory in search for food, going some 30-40 meters far from its nest (its path being hundreds of meters long). As it is a solitary insect, to go back home it cannot rely on pheromone traces left by swarm mates. However, after a winding exploration route, the way back home is strikingly straight.

It has been proved experimentally that when walking, the desert updates some kind of representation of the current direction and distance from its nest (the ant's orientation with respect to the sun plays a central role). This representation provides, with some approximation, the same kind of information that would be provided by a "nest sensor", should the desert ant have one In this case, we can regard the representation as a substitute for a sensor: the agent kind of "sees" through eyes it does not have, or at least it can "see" much further in space than its real eyes allow.