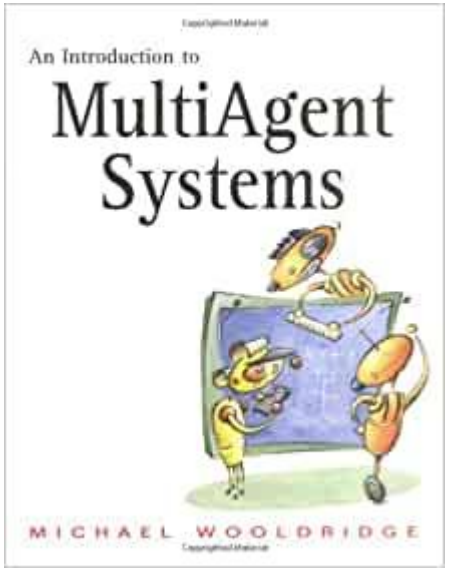
What is an agent?

Madalina Croitoru

University of Montpellier



Based on “An Introduction to MultiAgent Systems” by Michael Wooldridge, John Wiley & Sons, 2002.
http://www.csc.liv.ac.uk/~mjw/pubs/imas/

Programming progression…

- Programming has progressed through:
  - machine code;
  - assembly language;
  - machine-independent programming languages;
  - sub-routines;
  - procedures & functions;
  - abstract data types;
  - objects;

to *agents*.

Agents, a Definition

- An agent is a computer system that is capable of *independent* action on behalf of its user or owner (figuring out what needs to be done to satisfy design objectives, rather than constantly being told)
- To successfully interact, they will require the ability to *cooperate*, *coordinate*, and *negotiate* with each other, much as people do

Agent Design, Society Design

- Two key problems:
  - How do we build agents capable of independent, autonomous action, so that they can successfully carry out tasks we delegate to them?
  - How do we build agents that are capable of interacting (cooperating, coordinating, negotiating) with other agents in order to successfully carry out those delegated tasks, especially when the other agents cannot be assumed to share the same interests/goals?
- The first problem is *agent design*, the second is *society design* (micro/macro)

Autonomous Agents for specialized tasks

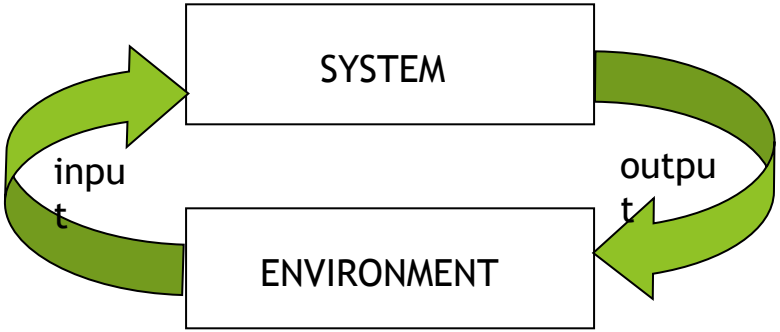
- When a space probe makes its long flight from Earth to the outer planets, a ground crew is usually required to continually track its progress, and decide how to deal with unexpected eventualities. This is costly and, if decisions are required *quickly*, it is simply not practicable. For these reasons, organizations like NASA are seriously investigating the possibility of making probes more autonomous — giving them richer decision making capabilities and responsibilities.
  - *This is not fiction:* NASA*'s DS1 has done it!*
- Agents (and their physical instantiation in robots) have a role to play in high-risk situations, unsuitable or impossible for humans
- The degree of autonomy will differ depending on the situation (remote human control may be an alternative, but not always)

Internet Agents

- Searching the Internet for the answer to a specific query can be a long and tedious process. So, why not allow a computer program — an agent — do searches for us? The agent would typically be given a query that would require synthesizing pieces of information from various different Internet information sources. Failure would occur when a particular resource was unavailable, (perhaps due to network failure), or where results could not be obtained.

What is an Agent?

- The main point about agents is they are *autonomous*: capable of acting independently, exhibiting control over their internal state
- Thus: *an* agent *is a computer system capable of* autonomous action *in some environment in order to meet its* design objectives



Proactiveness and social ability

- Reacting to an environment is easy (e.g., stimulus → response rules)
- But we generally want agents to *do things for us*
- Hence *goal directed behavior*
- Pro-activeness = generating and attempting to achieve goals; not driven solely by events; taking the initiative
- Recognizing opportunities: Some goals can only be achieved with the cooperation of others
  - *Social ability* in agents is the ability to interact with other agents (and possibly humans) via some kind of *agent-communication language*, and perhaps cooperate with others

Environments – *Accessible* vs. *inaccessible*

- An accessible environment is one in which the agent can obtain complete, accurate, up-to-date information about the environment's state
- Most moderately complex environments (including, for example, the everyday physical world and the Internet) are inaccessible
- The more accessible an environment is, the simpler it is to build agents to operate in it

Environments – *Deterministic* vs. *non-deterministic*

- A deterministic environment is one in which any action has a single guaranteed effect — there is no uncertainty about the state that will result from performing an action
- The physical world can to all intents and purposes be regarded as non-deterministic
- Non-deterministic environments present greater problems for the agent designer

Environments - *Static* vs. *dynamic*

- A static environment is one that can be assumed to remain unchanged except by the performance of actions by the agent
- A dynamic environment is one that has other processes operating on it, and which hence changes in ways beyond the agent's control
- Other processes can interfere with the agent's actions (as in concurrent systems theory)
- The physical world is a highly dynamic environment

- Assume the environment may be in any of a finite set E of discrete, instantaneous states:

$$E = \{e, e', \ldots\}.$$

- Agents are assumed to have a repertoire of possible actions available to them, which transform the state of the environment:

$$Ac = \{\alpha, \alpha', \ldots\}$$

- A *run*, r, of an agent in an environment is a sequence of interleaved environment states and actions:

$$r : e_0 \xrightarrow{\alpha_0} e_1 \xrightarrow{\alpha_1} e_2 \xrightarrow{\alpha_2} e_3 \xrightarrow{\alpha_3} \cdots \xrightarrow{\alpha_{u-1}} e_u$$

Purely Reactive Agents

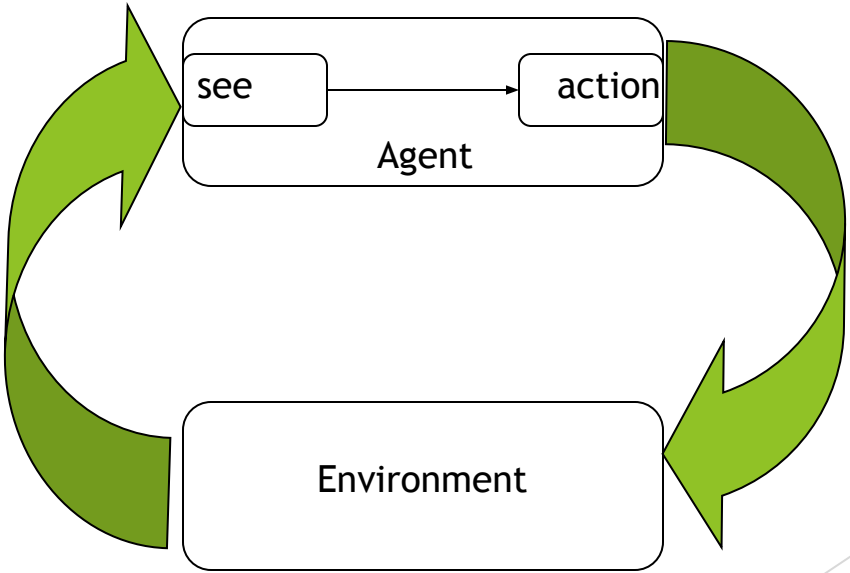
- Some agents decide what to do without reference to their history — they base their decision making entirely on the present, with no reference at all to the past
- We call such agents *purely reactive*:
- A **thermostat** is a purely reactive agent

$$action : E \rightarrow Ac$$

$$action(e) = \begin{cases} \text{off} & \text{if } e = \text{temperature OK} \\ \text{on} & \text{otherwise.} \end{cases}$$

Perception

- Now we can introduce a *perception* system:



Perception

- The *see* function is the agent's ability to observe its environment, whereas the *action* function represents the agent's decision making process
- ***Output*** of the *see* function is a ***percept***:

$$see : E \rightarrow Per$$

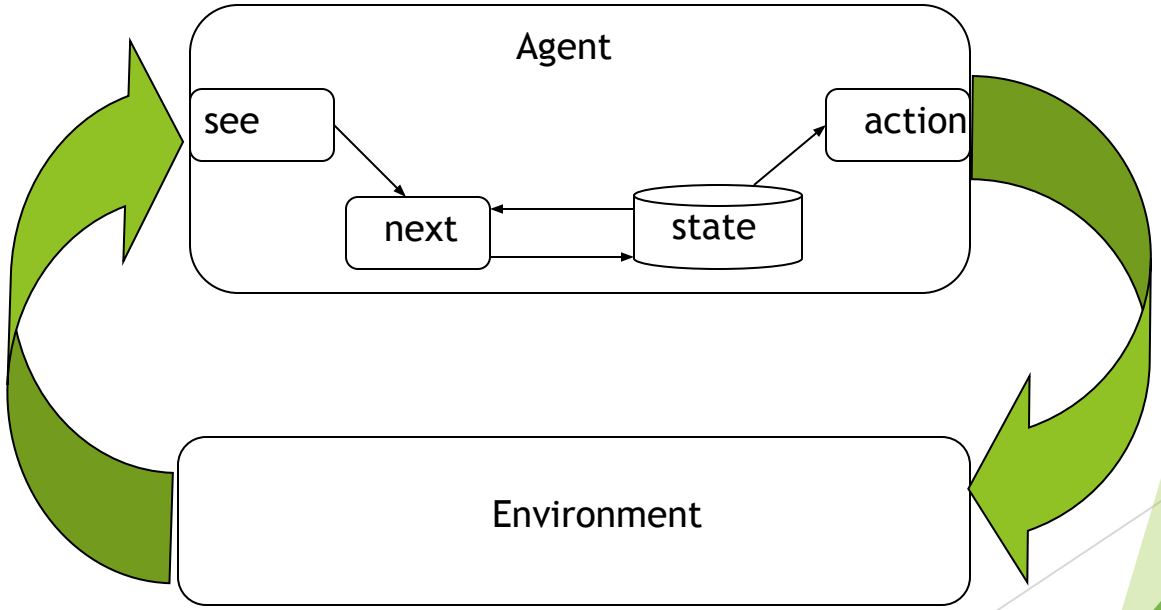
which maps environment states to percepts, and *action* is now a function

$$action : Per^* \rightarrow A$$

which maps sequences of percepts to actions

Agents with State

- We now consider agents that *maintain state*:



- These agents have some internal data structure, which is typically used to record information about the environment state and history.

Let I be the set of all internal states of the agent.

- The perception function see for a state-based agent is unchanged:

$$see : E \rightarrow Per$$

The action-selection function $action$ is now defined as a mapping

$$action : I \rightarrow Ac$$

from internal states to actions. An additional function $next$ is introduced, which maps an internal state and percept to an internal state:

$$next : I \times Per \rightarrow I$$

Utility Functions over states

- Associate *utilities* with individual states — the task of the agent is then to bring about states that maximize utility

But what is the value of a *run*…
- minimum utility of state on run?
- maximum utility of state on run?
- sum of utilities of states on run?
- average?

Disadvantage: difficult to specify a *long term* view when assigning utilities to individual states
(One possibility: a *discount* for states later on.)

Exercise for next week

- Give other (2-4) examples of agents (not necessarily intelligent) that you know of. For each define precisely:
  - The environment that the agents occupy
  - The states that the environment can be in
  - Is the environment accessible, deterministic or static?
  - The actions available to the agent (define their pre and post conditions)
  - The goal (or design objectives) of the agent
  - How / if the agents interact amongst each other
  - A few examples of runs
- If you have time you can start implementing one of the agent architectures above