Introduction to Multiagent Systems

Michal Jakob

Agent Technology Center, Dept. of Cybernetics, FEE Czech Technical University

A4M33MAS Autumn 2010 - Lect. 1

Lecture Outline

General Info

- Lecturers: **prof. Michal Pechoucek**, Michal Jakob and Peter Novak
- Tutorials: Peter Novak and Branislav Bosansky
- 12 lectures and 12 tutorials
- Course web page: `https://cw.felk.cvut.cz/doku.php/courses/a4m33mas/start`
- Recommended reading:
  - Russel and Norvig: Artificial Intelligence: Modern Approach
  - M. Wooldridge: An Introduction to MultiAgent Systems
  - J. M. Vidal: Multiagent Systems: with NetLogo Examples (available on-line)
  - Y. Shoham and K. Leyton-Brown: Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations (available on-line)
  - V. Marik, O. Stepankova, J. Lazansky a kol.: Umela inteligence (3)

Course Requirements/Grading

- Total 100 pts
  - Semestral project 1 (20 pts) – due midterm
  - Semestral project 2 (30 pts) – due end of term
  - Final exam (50 pts)
- At least 25 points required from each part

Lecture Outline

Trends in Computing

- **Ubiquity**: Cost of processing power decreases dramatically (e.g. Moore's Law), computers used everywhere
- **Interconnection**: Formerly only user-computer interaction, nowadays distributed/networked systems (Internet etc.)
- **Complexity**: Elaboration of tasks carried out by computers has grown
- **Delegation**: Giving control to computers even in safety-critical tasks (e.g. aircraft or nuclear plant control)
- **Human-orientation**: Increasing use of metaphors that better reflect human intuition from everyday life (e.g. GUIs, speech recognition, object orientation)

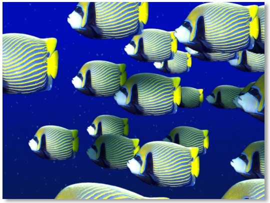
New Challenges for Computer Systems

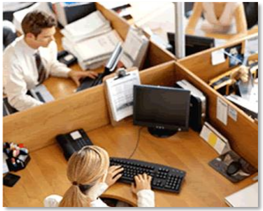
- Traditional design problem: *How can I build a system that produces the correct output given some input?*
  - Each system is more or less isolated, built from scratch
- Modern-day design problem: *How can I build a system that can operate independently on my behalf in a networked, distributed, large-scale environment in which it will need to interact with different other components pertaining to other users?*
  - Each system is built into an existing, persistent but constantly evolving computing ecosystem – it should be robust with respect to changes
  - No single owner and/or central authority
- In particular, distributed systems in which different components have different goals and need to cooperate have not been studied until recently

Multiagent Systems

- A field that has emerged as a consequence of the aforementioned problems
- Two fundamental ideas:
  - Individual agents are capable of autonomous action to a certain extent (they don't need to be told exactly what to do)
  - These agents interact with each other in multiagent systems (and which may represent users with different goals)
- Foundational problems of multiagent systems (MAS):
  1. The **agent design problem**: how should agents act to carry out their tasks?
  2. The **society design problem**: how should agents interact to carry out their tasks?
- These are known as the **micro** and **macro** perspective of MAS

Examples of Multiagent Systems


Animal herds/schools


Human teams and companies

Markets and economies

Transportation networks

Communication networks

Robotic teams

Multiagent Systems

Definition (Multiagent System)

Multiagent system is a collection of multiple autonomous agents, each acting towards its objectives while all interacting in a shared environment, being able to communicate and possibly coordinating their actions.

- Agents can be
  - **isolated** – equivalent to single-agent systems
  - **cooperative** – acting jointly towards a shared goal (team objective)
  - **self-interested** – each maximizing its own good while considering the other agents
  - combination of the above – cooperating with some, competing with others
- MAS $\sim$ **“socially-inspired computing”**
- Top-down rather than bottom-up construction

Applications of MAS

- Typical problems
  - Non-cooperative setting with self-interested agents unwilling to cooperate
  - Distributed setting with inability (or ineffectiveness) to create and maintain shared global knowledge
  - Highly dynamic environments where fast reaction and frequent replanning is necessary
- **manufacturing and logistics** – production planning, inventory management, supply chain/network management, procurement
- **markets** – automated trading/auctioning, auction mechanism analysis and design, strategy modeling, market modeling
- **ubiquitous computing** – context-enabled personal assistance, user modeling, privacy management, power-consumption optimization
- **robotic teams** – team planning and coordination, optimum team composition, coalition formation, information fusion

Applications of MAS

- **utility networks** – smart grid management, smart appliances, consumption pattern modeling
- **computer and communication networks** – load balancing, intrusion detection, bandwidth management, monitoring
- **transportation** – intelligent road infrastructure, cooperative driving, congestion reduction
- **security and defense** – mission planning and execution, optimum patrolling and surveillance, opponent modeling, vulnerability assessment
- **computer games and computer animation** - game AI, behavioral animation, NPC implementation
- **policy-making support** – modeling of various socio-economical phenomena (crime, migration, urban growth)
- . . . new applications appearing in the increasingly interconnected world

Topics in Multiagent Systems

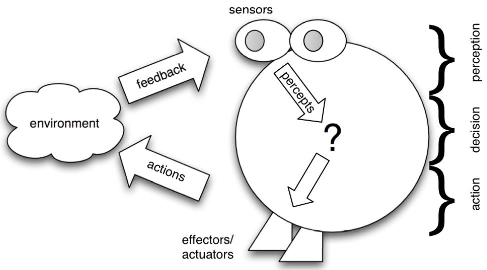
- How should agent's objectives be specified?
- How should agent's control logic be implemented so that the agents acts towards its objectives?
- What language should agents use to communicate their beliefs and aspiration?
- Which protocols should agents use to negotiate and agree/choose if there are multiple options (as there always are)?
- How should agents in a team decompose and allocate tasks so as to effectively achieve team's common goal?
- How should the agent maximize it utility in the presence of other competing and possible hostile agents?
- Which voting mechanisms are robust against manipulation?
- How does ordered behavior emerge from seemingly chaotic interactions?
- . . .

Course Schedule

1. Introduction to multiagent systems
2. Agent architectures – reactive, deliberative and BDI architecture
3. Tools for programming intelligent agents; agent programming languages Jason, AgentSpeak
4. Distributed problem solving, distributed constraint optimization
5. Non-cooperative game theory, games in normal form, prisoner's dilemma, Nash equilibrium, solution concepts
6. Extensive form games, game tree search
7. Bilateral negotiation, auctions and auction protocols
8. Combinatorial auctions, mechanism design, voting
9. Cooperative game theory, coalition formation
10. Modeling agents in formal logic: model logic, modeling time, action, joint knowledge
11. Modeling agents in formal logic: reasoning about commitments, formal model of the BDI architecture
12. Development of scalable multiagent systems, applications

Lecture Outline

What is Agent?



Definition (Russell & Norvig)

An agent is anything that can perceive its environment (through its sensors) and act upon that environment (through its effectors)

.

- Focus on **situatedness** in the environment (**embodiment**)
- The agent can only influence the environment but not fully control it (sensor/effector failure, non-determinism)

What is Agent? (2)

Definition (Wooldridge & Jennings)

An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives/delegated goals

- Adds a second dimension to agent definition: the relationship between agent and designer/user
  - Agent is capable of independent action
  - Agent action is purposeful
- Autonomy is a central, distinguishing property of agents

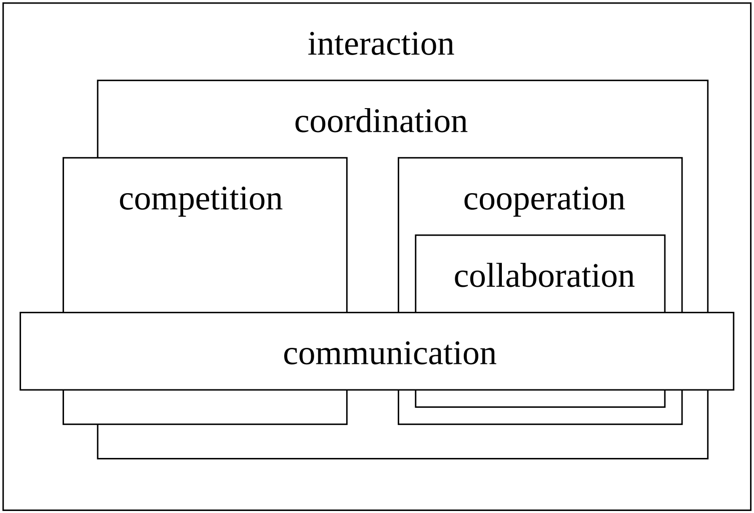
Agent Properties

- **autonomous** – the agent is self goal-directed and acts without requiring user initiation and guidance; it can choose its own goal and the way to achieve it; its behavior is determined by its experience; we have no direct control over it
- **reactive** – the agent maintains an ongoing interaction with its environment, and responds to changes that occur in it
- **proactive** – the agent generates and attempts to achieve goals; it is not driven solely by events but takes the initiative

Agent Properties

- **sociable** – the agent interacts with other agents (and possibly humans) via cooperation, coordination, and negotiation; it is aware and able to reason about other agents and how they can help it achieve its own goals
  - **coordination** is managing the interdependencies between actions of multiple agents (not necessarily cooperative)
  - **cooperation** is working together as a team to achieve a shared goal
  - **negotiation** is the ability to reach agreements on matters of common interest
- Systems of the future will need to be good at teamwork

Typology of Interaction



Lecture Outline

Agent Behavior

- Agent's behavior is described by the **agent function** that maps percept sequences to actions

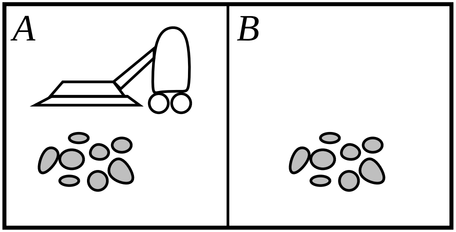
$$f : \mathscr{P} \mapsto \mathscr{A}$$

- The **agent program** runs on a physical architecture to produce f
- Key questions: What is the *right* function? Can it be implemented in a small agent program?

Agent Behavior

- Agent's behavior is described by the **agent function** that maps percept sequences to actions
$$f : \mathscr{P} \mapsto \mathscr{A}$$
- The **agent program** runs on a physical architecture to produce f
- Key questions: What is the *right* function? Can it be implemented in a small agent program?

Example: Vacuum Cleaner World



- Percepts: location and contents, e.g. $[A, Dirty]$
- Actions: *Left*, *Right*, *Suck*, *NoOp*

Vacuum Cleaner Agent

Percept sequence	Action
$[A, Clean]$	*Right*
$[A, Dirty]$	*Suck*
$[B, Clean]$	*Left*
$[B, Dirty]$	*Suck*
$[A, Clean], [A, Clean]$	*Right*
$[A, Clean], [A, Dirty]$	*Suck*
$\vdots$	$\vdots$
$[A, Clean], [A, Clean], [A, Clean]$	*Right*
$[A, Clean], [A, Clean], [A, Dirty]$	*Suck*
$\vdots$	$\vdots$

Rational Behavior

What is the right behavior?

Definition (Rational Agent)

Rational agent chooses whichever action **maximizes the expected value of the performance measure** given the percept sequence to date and whatever bulit-in knowledge the agent has.

- Rationality is relative and depends on four aspects:
  1. performance measure which defines the degree of success
  2. percept sequence (complete perceptual history)
  3. agent's knowledge about the environment
  4. actions available to the agent
- Rational $\neq$ omniscient, rational $\neq$ clairvoyant $\Rightarrow$ rational $\neq$ successful

Specifying Task Environments

To design a rational agent, we must specify the **task environment (PEAS)**

1. Performance measure
2. Environment
3. Actuators
4. Sensors

Task environments define problems to which rational agents are the solutions.

PEAS Examples

Agent	Performance measure	Environment	Actuators	Sensors
Taxi driver	safe, fast, legal, comfortable trip, maximize profits	roads, other traffic, pedestrians, customers	steering, accelerator, brake, signal, horn, display	cameras, sonar, speedometer, GPS, engine sensors, keyboard
Part picking robot	percentage of parts in correct bins	conveyor belt with parts, bins	jointed arm and hand	camera, joint angle sensors
Trading agent	maximum profit over a defined period	electronic trading platform	API for placing trading orders	current and historic prices, current orders
Refinery controller	maximize purity, yield, safety	refinery operators	valves, pumps, heaters, displays	temperature, pressure, chemical sensors

Properties of Environments

- **Fully observable vs. partially observable** – can agents obtain complete and correct information about the state of the world?
- **Deterministic vs. stochastic** – Do actions have guaranteed and uniquely defined effects?
- **Episodic vs. sequential** – Can agents decisions be made for different, independent episodes?
- **Static vs. dynamic** – Does the environment change by processes beyond agent control?
- **Discrete vs. continuous** – Is the number of actions and percepts fixed and finite?
- **Single-agent vs. multi-agent** – Does the behavior of one agent depends on the behavior of other agents?

Example Environments

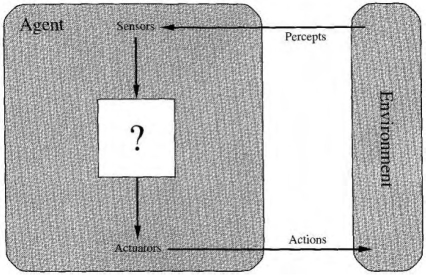
	Solitaire	Backgammon	Internet shopping	Taxi
Observable	No	Yes	No	No
Deterministic	Yes	No	Partly	No
Episodic	No	No	No	No
Static	Yes	Semi	Semi	No
Discrete	Yes	Yes	Yes	No
Single-agent	Yes	No	Yes (except auctions)	No

Lecture Outline

Implementing the Agent

How should one implement the agent function?

- So that the resulting behavior is (near) rational.
- So that its calculation is computationally tractable.



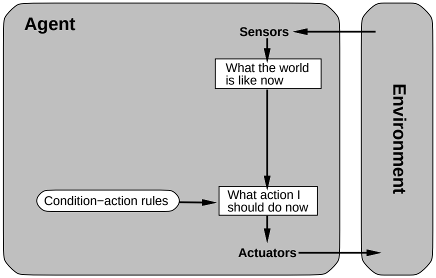
Hierarchy of Agents

Four basic types of agent in the order of increasing capability:

1. simple reflex agents
2. reflex agents with state
3. goal-based agents
4. utility-based agents

All these can be turned into learning agents.

Simple Reflex Agents



Simple reflex agent chooses the next action on the basis of the current percept

- Condition-action rules provide a way to present common regularities appearing in input/output associations
- Ex.: `if car-in-front-is-braking then initialize-braking`

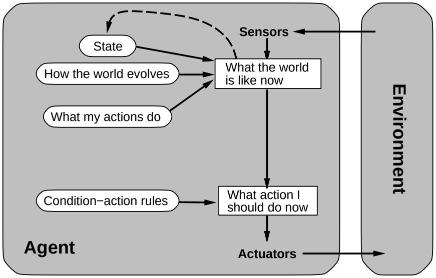
Adding State

Decision making is seldom possible based on the basis of a single percept

- the choice of action may depend on the entire percept history
- sensors do not necessarily provide access to the complete state of the environment

$\Rightarrow$ It can be advantageous to store information about the world in the agent.

Reflex Agents with State

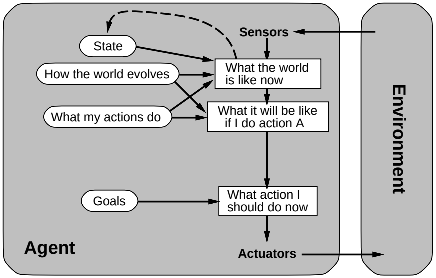


Reflex agent with internal state keeps track of the world by extracting relevant information from percepts and storing it in its memory.

Telling the Agent What to Do

- Previous types of agents have the behavior hard-coded in their rules – there is no way to tell them what to do
- Fundamental aspect of autonomy: we want to tell agent what to do but not how to do it!
- We can specify
    - action to perform – *not interesting*
    - (set of) goal state(s) to be reached $\rightarrow$ goal-based agents
    - a performance measure to be maximized $\rightarrow$ utility-based agents

Goal-based Agents


Sensors
State
How the world evolves
What my actions do
What the world is like now
What it will be like if I do action A
Goals
What action I should do now
Agent
Actuators
Environment

- **Problem**: goals are not necessarily achievable by a single action:
  - $\rightarrow$ **search and planning** are subfields of AI devoted to finding actions sequences that achieve the agent's goals.
- Goal-based agent utilizes goals and planning to determine which action to take.

Towards Utility-based Agents

Goals alone are not sufficient for decision making:

1. there may be multiple ways of achieving them;
2. agents may have several conflicting goals that cannot be achieved simultaneously.

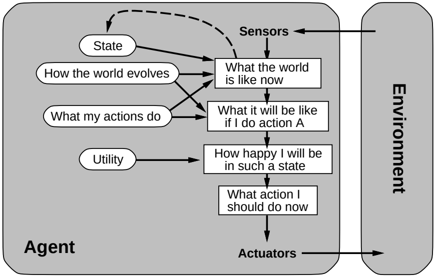
We introduce the concept of **utility**:

- utility is a function that maps a state onto a real number; it captures "quality" of a state
- if an agent prefers one world state to another state then the former state has higher utility for the agent.

Utility can be used for:

1. choosing the best plan
2. resolving conflicts among goals
3. estimating the successfulness of an agent if the outcomes of actions are uncertain

Utility-based Agents


Sensors
State
How the world evolves
What my actions do
Utility
What the world is like now
What it will be like if I do action A
How happy I will be in such a state
What action I should do now
Agent
Actuators
Environment

Utility-based agent use the utility function to choose the most desirable action/course of actions to take.

Summary

- Multiagent systems approach ever more important in the increasingly interconnected world where systems are required to cooperate flexibly
  - "society-inspired computing"
- Intelligent agent is autonomous, proactive, reactive and sociable
- Agents can be cooperative or self-interested (or combination thereof)
- There are different agent architectures with different capabilities and complexity
- Related reading:
  - Russel and Norvig: Artificial Intelligence: A Modern Approach – Chapter 2
  - Wooldrige: An Introduction to Multiagent Systems – Chapters 1 and 2