INTRODUCTION

1. Overview

2. The Vision Thing

3. Some Views of the Field

- **Agents as a paradigm for software engineering**
- **Agents as a tool for understanding human societies**

4. Objections to Multiagent Systems

- **Is it not all just distributed/concurent systems?**
- **Is it not all just artificial intelligence (AI)?**
- **Is it not all just economics/game theory?**
- **Is it not all just social science?**

Overview

- Five ongoing trends have marked the history of computing:
  - *ubiquity*;
  - *interconnection*;
  - *intelligence*;
  - *delegation*; and
  - *human-orientation*

Ubiquity

- The continual reduction in cost of computing capability has made it possible to introduce processing power into places and devices that would have once been uneconomic
- As processing capability spreads, sophistication (and intelligence of a sort) becomes ubiquitous
- What could benefit from having a processor embedded in it…?

Interconnection

- Computer systems today no longer stand alone, but are networked into large distributed systems

- The internet is an obvious example, but networking is spreading its ever-growing tentacles…

Intelligence

- The complexity of tasks that we are capable of **automating** and **delegating** to computers has grown steadily

Delegation

- Computers are doing more for us – without our intervention
- We are *giving control* to computers, even in safety critical tasks

- One example: fly-by-wire aircraft, where the machine's judgment may be trusted more than an experienced pilot

Human Orientation

- The movement away from machine-oriented views of programming toward concepts and metaphors that more closely reflect the way we ourselves understand the world
- Programmers (and users!) relate to the machine differently
- Programmers conceptualize and implement software in terms of higher-level – more human-oriented – abstractions

Programming progression…

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

to *agents*.

Global Computing

- What techniques might be needed to deal with systems composed of 10^{10} processors?

- Hundreds of millions of people connected by email once seemed to be “science fiction”…
- Let’s assume that current software development models can’t handle this…

Where does it bring us?

- **Delegation** and **Intelligence** imply the need to build computer systems that can act effectively on our behalf
- This implies:
  - The ability of computer systems to act *independently*
  - The ability of computer systems to act in a way that *represents our best interests* while interacting with other humans or systems

Interconnection and Distribution

- Interconnection and Distribution have become core motifs in Computer Science
- But Interconnection and Distribution, coupled with the need for systems to represent our best interests, implies systems that can *cooperate* and *reach agreements* (or even *compete*) with other systems that have different interests (much as we do with other people)

So Computer Science expands…

- These issues were not studied in Computer Science until recently
- All of these trends have led to the emergence of a new field in Computer Science: *multiagent systems*

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)

Multiagent Systems, a Definition

- A multiagent system is one that consists of a number of agents, which *interact* with one-another
- In the most general case, agents will be acting on behalf of users with different goals and motivations
- To successfully interact, they will require the ability to *cooperate*, *coordinate*, and *negotiate* with each other, much as people do

Agent Design, Society Design

- The course covers 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)

Multiagent Systems

- In Multiagent Systems, we address questions such as:
  - How can cooperation emerge in societies of self-interested agents?
  - What kinds of languages can agents use to communicate?
  - How can self-interested agents recognize conflict, and how can they (nevertheless) reach agreement?
  - How can autonomous agents coordinate their activities so as to cooperatively achieve goals?