Introduction to Intelligent Agents

Pınar Yolum
p.yolum@uu.nl

Utrecht University
Course Information

• Jointly taught with Mehdi Dastani
• Topics
• Work Schedule
• Grading
• Resources
• Academic Integrity
Project Domains

- Shopping (e.g., an agent to help its user buy things she needs)
- Cinema (e.g., an agent to help its user to learn about genres or choose movies to go to)
- Medicine (e.g., an agent to help its user to diagnose an illness)
- Sport (e.g., an agent to answer the questions of a user about different sports)
- Cooking (e.g., an agent to help its user choose a recipe based on existing ingredients)
- Programming (e.g., an agent to help its user pick a programming language for a given task)
- Housing (e.g., an agent to help its user choose an apartment given various constraints)
- Holiday (e.g., an agent to help its user pick a location for holiday based on the constraints of a family)
Agents

Based largely on

Service-Oriented Computing: Semantics, Processes, Agents
– Munindar P. Singh and Michael N. Huhns, Wiley, 2004

and

Artificial Intelligence: A Modern Approach
Stuart Russell and Peter Norvig, 1995
Outline

• So, what are agents?
• Agent characteristics
• Agent environments
• Agent types
• Agent architectures
Agents

• An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
• Human agent: eyes, ears, and other organs for sensors; hands,
  • legs, mouth, and other body parts for actuators
• Robotic agent: cameras and infrared range finders for sensors;
  • various motors for actuators
Intelligent Agents

A persistent computation that can perceive, reason, act and communicate (Singh&Huhns)

• Agents usually represent some principals (people, businesses, etc.)
• Principals (and thus the agents) may have contradicting preferences, goals, etc.
• To cooperate, they will need to communicate, coordinate their actions, negotiate their goals, etc.
• Ex. Your agent talks to Amazon.com agent to settle on a price for a travel book about China
• Different principals, different goals?
MAS Example

Buyer (Tom) → Order → Seller (Amazon) → Schedule → Deliverer (UPS)

Buyer (Bob) → Order → Seller (Bol) → Schedule → Deliverer (DHL)

Buyer (Mary) → Order → Seller (B&N)

Marketplace (E-bay)
Agent vs. Object

• Communicate
  – Request agents to perform an action
  – Call methods of objects

• Reason
  – Agents reason based on the state of the world and perform actions
  – Objects always do what they are told

• Adapt
  – Agents can learn to act differently in different circumstances
  – Objects do not change their behavior over time
Agent Characteristics (1)

• Autonomous (Execution)
  – Freedom to act independently
  – Choose its own actions
  – Decide on other agents that it will interact with

• Heterogeneous (Design)
  – Could be designed and implemented by different parties
  – No need to expose internal properties
  – Might have to comply with some common requirements
Agent Characteristics (2)

• Adaptive
  – Agents can learn what is good for them (and who is useful for them) and change their behavior over time

• Self-interested
  – Agents work for their owners
  – No need to carry out a task because someone asks for it
Perspectives on Agents

• Artificial Intelligence:
  – Cognitive entity, possibly with feelings, emotions, etc.

• Database:
  – Information sources
  – Cooperate to carry out transactions

• Desktop software:
  – Proxy for a computation; mostly an interface
  – SNMP agents; Microsoft agent
Open Environment

- Autonomous entities
  - Can enter and leave frequently
  - Don’t need to let anyone know
  - No control over what others will do

- Cooperative entities
  - No use of sellers if they cannot cooperate with customers
  - Need for rules, regulations, contracts
Generic Agent Architecture (1)

• Assume that the environment can be in one of the following states \( S: \{s_0, s_1, s_2, \ldots s_n\} \)

• The set of actions that an agent can do is

\[ A: \{a_0, a_1, a_2, \ldots a_n\} \]

• Deterministic behavior: Takes a sequence of states and determines what actions to take.

\[ \text{action: } S^* \rightarrow A \]

– Actions include communicative actions

• Nondeterministic effect: Takes a state and an action and determines the possible set of states

\[ \text{environment: } S \times A \rightarrow \varphi\{S\} \]
Generic Agent Architecture (2)

• History: Sequence of environment states and agent actions

\[ h: s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \ldots s_n \]

• Short-lived computations: \( n \) is fixed; agent eventually terminates

• Long-lived (persistent) computations: Loop forever

• Full history: If the agent remembers all the states it has been in as well as all its actions
A Rational Agent

Rationality depends on...

- The performance measure for success
- What the agent has perceived so far
- What the agent knows about the environment
- The actions the agent can perform

An **ideal rational agent**: *for each possible percept sequence, it acts to maximize its expected utility, on the basis of its knowledge and the evidence from the percept sequence*
• Agents are situated in an environment
• Different environments (based on Russell and Norvig)
• Accessible environment:
  – If an agent can perceive everything happening around;
  – Agents have complete information.
  – Rarely happens in practice
  – Inaccessible environments: Internet, physical world
Environment (2)

• Deterministic environment:
  – Each action has a guaranteed effect
  – An agent can determine the state of the world by knowing the state before an action happens and knowing the effect of the action
  – Physical world: Deterministic?
Environment (3)

• History freedom:
  – Episode: Single cycle of an agent perceiving and taking an action
  – Episodic: If the choice depends on the current episode and not on previous episodes
    • Easier to operate
  – Sequential: History matters
    • Needs thinking ahead
    • Chess
Environment (4)

• Static environment:
  – The environment only changes by the actions of the agent
  – Ex: Chess

• Dynamic environment:
  – Other agents’ actions can change the environment
  – The environment itself can change over time
  – Ex: Internet
Environment (5)

- **Discrete environment:**
  - Fixed, finite number of actions and percepts
  - Chess: At each time point, the number of actions is finite
- **Continuous environment:**
  - Not composed of discrete units
  - Either number of actions or percepts infinite
- Possible to convert continuous environments into discrete environments (with loss of precision)
• Single vs. Multiagent

• Open environment:
  – Autonomous entities
    • Can enter and leave frequently
    • Don’t need to let anyone know
    • No control over what others will do
  – Cooperative entities
    • No use of sellers if they cannot cooperate with customers
    • Need for rules, regulations, contracts
## Environment types

<table>
<thead>
<tr>
<th></th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet Shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable??</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deterministic??</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Episodic??</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Static??</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discrete??</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single-agent??</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Environment types

<table>
<thead>
<tr>
<th>Observable??</th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet Shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Deterministic??</td>
<td>Yes</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td>Episodic??</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Static??</td>
<td>Yes</td>
<td>Semi</td>
<td>Semi</td>
<td>No</td>
</tr>
<tr>
<td>Discrete??</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Single-agent??</td>
<td>Yes</td>
<td>No</td>
<td>Yes (except auctions)</td>
<td>No</td>
</tr>
</tbody>
</table>
Agent Architectures

- Reactive
- Goal-Based
- Utility-Based
- Belief-Desire-Intention (BDI)
- Layered Architecture
Simple reflex agents
Reactive Agents (1)

• Stateless reactive agents:  \( \text{action}: S \rightarrow A \)
  
  – Don’t care about the history
  
  – Act based on the current state now

• Famous thermostat example!
  
  – Perception allows the agent to get the temperature
  
  – The agent compares the temperature to a range of values to decide if it is OK

\[
\text{action}(S) = \begin{cases} 
\text{heater off if temperature = OK} \\
\text{heater on otherwise}
\end{cases}
\]

• Simple to implement; a set of condition-action rules

• Cannot be applied if the history is important
Simple Reactive Agent

- Select actions based only on the current percept, ignoring the rest of the percept history.
- Table lookup of percept-action pairs defining all possible condition-action rules necessary to interact in an environment
- Problems
  - Too big to generate and to store (Chess has about $10^{120}$ states, for example)
  - No knowledge of non-perceptual parts of the current state
  - Not adaptive to changes in the environment; requires entire table to be updated if changes occur
  - Looping: Can't make actions conditional
Model-based reflex agents (with state)
• The knowledge about “how the world works” is called a **model** of the world.

• An agent that uses such a model is called a **model-based agent**.

• Encode "internal state" of the world to remember the past as contained in earlier percepts

• Needed because sensors do not usually give the entire state of the world at each input, so perception of the environment is captured over time. "State" used to encode different "world states" that generate the same immediate percept.

• Requires ability to represent change in the world; one possibility is to represent just the latest state, but then can't reason about hypothetical courses of action
Reactive Agents (2)

• Add an internal state
  – Start with an initial internal state
  – Perceive the environment and update the internal state accordingly
  – Act based on the internal state

• Still simple to implement; need more memory for the state and a set of condition-action rules
Reactive Architecture

- Does not rely on symbol manipulation, e.g., Rodney Brooks’ subsumption architecture
- Intelligent behavior can be generated without explicit representations proposed by symbolic AI
- Intelligent behavior can be generated without explicit abstract reasoning
- Intelligence is an emergent property of certain complex systems
Subsumption Architecture

• A hierarchy of task-accomplishing behaviors
• Each behavior is a rather simple rule-like structure
• Each behavior competes with others to exercise control over the agent
• Lower layers present more primitive kinds of behavior
• In terms of computation, the resulting systems are extremely simple
Goal-Oriented Agents

• Add a goal
  – What is the agent trying to do?
  – Drive to Taksim; buy a book online
  – Perform actions that will realize the goal

• A set of condition-action rules is not enough

• Think about a goal state where the desired goal holds

• Plan or search a sequence of actions such that applying those actions will transform the current state into the goal state
Goal-based agents (R&N)
Utility-Based Agents (1)

• A goal-oriented agent may achieve its goals in a number of ways.
• Nothing is said about whether one way is better than another
• However, one path can be preferred over another because it’s shorter, cheaper, etc.
• Utility:
  – Quantitative measure of a chosen path
  – A function from states to real numbers
Utility-Based Agents (2)

• Agents may have conflicting goals*
  – Buy a book with a low cost and buy it fast
  – Which goal should have precedence over the other?

• Assign a utility to each state

• Maximize expected utility
Utility-based agents (R&N)
BDI Agents

• Agents are represented with beliefs, desires, and intentions.

• Beliefs:
  – What the agent thinks is true in the environment
  – Need not be correct

• Desires:
  – The environment that the agent prefers to exist in
  – Can be inconsistent
• Intentions:
  – The environment that the agent is trying to achieve
  – Subset of agent’s desires
  – Should be consistent with each other
  – Agent’s actions are related to its intentions
  – Agent carries out a plan to realize an intention
Means-end Reasoning

• Decide what intentions to adopt or revise
• What actions to perform
• The agent does multiple things concurrently
  – An agent thread constantly checks the environment
  – An agent thread uses the environment input to revise its internal working
  – Possibly more threads for communication, acting, etc.
Revision

• Action revision
  – Check if the state of the environment is compatible with its actions
  – If they are not compatible, revise actions

• Example:
  – Intention to be at the university at 10:00 am
  – Perceive from the environment that it is already 10:15

• Realizing an intention
  – Carrying out a plan of actions
  – Modified intention means modified plan

• Incompatible environment and intention
  – Drop intention (Too late to go)
  – Revise intention (Try to be there at 10:30)
Revision

• Belief revision
  – Perceive a new belief that contradicts old belief
• Example:
  – Belief: Course starts at 10:00
  – Talk to a friend and learn that the course starts at 11:00
• Need to resolve it
  – Ask other people; look up on the Web
  – Rules for belief revision
    • New belief overrides an inconsistent old belief
Architecture of BDI-Based Agent

Execution Cycle:
1. New information arrives that updates beliefs and goals
2. Actions are triggered by new beliefs or goals
3. A triggered action is intended
4. An intended action is selected
5. That intention is activated
6. An action is performed
7. New beliefs or goals are stored
8. Intentions are updated
Knowledge (1)

• Knowledge=“True” belief?
• Knowledge representation for reasoning
  – Expressed in machine-understandable form
  – Logic (Description Logic, Propositional Logic, FOL)
• Two aspects:
  – Syntax: Describe what a sentence is
    • Allowed sequences of characters, words, etc.
  – Semantic: Determines the facts that the sentence corresponds to
• Example: $x \leq y$
• Need precise syntax and semantics to reason
Knowledge (2)

- Facts in real world vs. representation in agents
- Reasoning derives new knowledge
- Reasoning on facts and representation should yield the same knowledge
- Entailment: Deriving new sentences based on old ones
  \[ \text{KB} \models \alpha \]
- Inference Procedure
  - Generate all sentences entailed by KB
  - Check if alpha is entailed by KB
- Sound: If the procedure generates entailed sentences only
- Complete: If the procedure has a proof for every entailed sentence
Logic-Based Agency

• Decision making is realized through logical deduction
• View the agents as particular type of knowledge based system
  – Contains an explicitly represented symbolic model of the world
  – Takes decisions via symbolic reasoning
• Apply inference rules to derive consequences
  – Proving that an intention can no longer be achieved
  – Deciding on what new action to take
• Problems:
  – Translating the real world into an accurate adequate symbolic description, in real-time
  – How to represent information symbolically about complex real-world entities
Layered Architecture

• Decompose the system into different layers to deal with different types of behaviors
• Typically at least two layers – to deal with reactive and proactive behaviors
• Broadly, two types of control flow within layered architectures
  • Horizontal layering
  • Vertical layering
Layered Architecture

Horizontal Layering

Vertical Layering (one pass control)

Vertical Layering (two pass control)
Developing Agents (1)

• Depending on what is expected of the agent
• An agent that interacts with other agents over the Web
  – JAVA!
  – Better support for Web standards
  – Easy integration with other programming environments
• An agent that processes logical formulae (logic-based agents)
  – Prolog
  – Java+Prolog
Developing Agents (2)

• Agent0, GOLOG, 3APL
  – Agent programming languages
  – Constructs for beliefs, goals, etc.
  – Specify rules to reason on these (update, delete)
  – Use an interpreter which will track which rules are fired and make modifications accordingly
Developing Agents (3)

• Java Agent DEvelopment Framework (JADE)
  – FIPA-compliant
  – Available libraries for agent templates

• MadKit
  – Agents programmed in Java
  – Enforce an organization model so that each agent plays a role in the MAS

• OAA, JATLite, IBM Able