Artificial Intelligence Foundation – JC3001

Lecture 3: Agents II

Prof. Aladdin Ayesh (aladdin.ayesh@abdn.ac.uk)

Dr. Binod Bhattarai (binod.bhattarai@abdn.ac.uk)

Dr. Gideon Ogunniye, (g.ogunniye@abdn.ac.uk)

September 2025

Material adapted from:
Russell and Norvig (AIMA Book): Chapter 2
Michael Wooldridge (An Introduction to Multi-Agent Systems)

UNIVERSITY OF ABERDEEN

Course Progression

- Part 1: Introduction
  1. Introduction to AI ✓
  2. **Agents**
- Part 2: Problem-solving
  1. Search 1: Uninformed Search
  2. Search 2: Heuristic Search
  3. Search 3: Local Search
  4. Search 4: Adversarial Search
- Part 3: Reasoning and Uncertainty
  1. Reasoning 1: Constraint Satisfaction
  2. Reasoning 2: Logic and Inference
  3. Probabilistic Reasoning 1: BNs
  4. Probabilistic Reasoning 2: HMMs
- Part 4: Planning
  1. Planning 1: Intro and Formalism
  2. Planning 2: Algos and Heuristics
  3. Planning 3: Hierarchical Planning
  4. Planning 4: Stochastic Planning
- Part 5: Learning
  1. Learning 1: Intro to ML
  2. Learning 2: Regression
  3. Learning 3: Neural Networks
  4. Learning 4: Reinforcement Learning
- Part 6: Conclusion
  1. Ethical Issues in AI
  2. Conclusions and Discussion

Objectives

- Motivate the abstraction of agents ✓
- Introduce the notion of intelligent agent
- Describe the various types of environment wherein an agent operates
- Describe various types of internal workings of an agent

Outline

An intelligent agent is a computer system capable of flexible autonomous action in some environment

- Typically intelligent agents exhibit 3 types of behaviour:
  - Reactive
  - Pro-active and
  - Social

- If a program's environment is guaranteed to be fixed, a program can just execute blindly
  Example of fixed environment: compiler
- The real world is not like that: most environments are dynamic
- A reactive system is one that maintains an ongoing interaction with its environment, and responds to changes that occur in it (in time for the response to be useful); i.e. in real time

- Reacting to an environment is easy; e.g. stimulus $\rightarrow$ response rules
- But we generally want agents to do things for us
- Hence goal-directed behaviour
- Pro-activeness:
  - Generating and attempting to achieve goals
  - Not driven solely by events
  - Taking the initiative
  - Recognising opportunities

Balancing Reactive and Goal-Oriented Behaviour

- We want our agents to be reactive, responding to changing conditions in an appropriate (timely) fashion
- We want our agents to systematically work towards long-term goals
- These two can be at odds with one another
- Designing an agent that can balance the two remains an open research problem

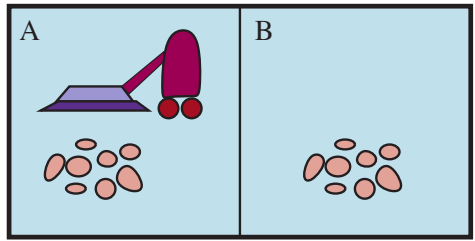
- The real world is a **multi**-agent environment: we must consider others when planning for goals
- Some goals can only be achieved with the cooperation of others
- Similarly for many computer environments (e.g., the Internet)
- **Social ability** in agents is the ability to interact with other agents (and possibly humans) via some kind of agent-communication language, and perhaps cooperate with others

- Objects:
  - Encapsulate state;
  - Communicate via message passing;
  - Have methods that can perform operations on its state
- Agents:
  - Autonomous: stronger notion of autonomy than objects, they decide whether or not to perform an action on request
  - Smart: capable of flexible (reactive, pro-active, social) behaviour;
  - Active: no passive service providers

- Classical Artificial Intelligence (AI) focusses on
  - Algorithms to solve hard decision problems; e.g. planning
  - Algorithms to interpret complex data; e.g. Machine Vision
  - Knowledge representation and reasoning
  - Applications in decision support, game playing
- Agents often combine these techniques
  - Robotics: vision + planning
  - Software assistants: machine learning + scheduling
  - Game 'bots': search + machine learning
- Don't we need to solve AI to build agents?
- Oren Etzioni (NetBot Inc.): We made our agents dumber and dumber and dumber...until finally they made money (seminar QMUL, 1998, shortly after NetBot Inc. was bought for $35M)

Vacuum World

- Two rooms
  Each room might be dirty or clean
- One vacuum robot
  Can move between rooms or suck



Fixed **performance measure** evaluates the environment sequence:

- one point per square cleaned up in time T?
- one point per clean square per time step, minus one per move?
- penalize for $> k$ dirty squares?

A **rational agent** chooses whichever action maximises the **expected** value of the performance measure, **given the percept sequence to date**.

- Rational $\neq$ omniscient: percepts may not supply all relevant information
- Rational $\neq$ clairvoyant: action outcomes may not be as expected
- Hence, rational $\neq$ successful
- Rational $\Rightarrow$ exploration, learning, autonomy

Outline

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

- Performance measure
- Environment
- Actuators
- Sensors

Environment type dimensions

- Observability
- Determinism
- Dynamicity
- Discreteness

Environments
Accessible vs. Inaccessible

- Accessibility is also referred to as **Observability** (Full to partial)
- An **accessible** or **fully observable** 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** or **partially observable**
- The more accessible an environment is, the simpler it is to build agents to operate in it

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

- In an **episodic** environment, the performance of an agent is dependent on a number of discrete episodes, with no link between the performance of an agent in different scenarios
- Episodic environments are simpler from the agent developer's perspective
- The agent can decide what to do based **only on the current episode**
- No need to reason about interactions between episodes

- 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 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 highly dynamic

- An environment is **discrete** if there are a **fixed**, finite number of actions and percepts in it
- **Continuous** environments have a certain level of mismatch with computer systems
- Discrete environments could in principle be handled by a kind of “lookup table”

Let's classify the following environments:

- Crossword puzzle
- Chess with a clock
- Poker
- Backgammon
- Taxi Driving
- Medical Diagnosis
- Image Analysis
- Part-picking robot
- Refinery controller
- Interactive English tutor

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword puzzle						
Chess with a clock						
Poker						
Backgammon						
Taxi Driving						
Medical Diagnosis						
Image Analysis						
Part-picking robot						
Refinery controller						
Interactive English tutor						

UNIVERSITY OF ABERDEEN

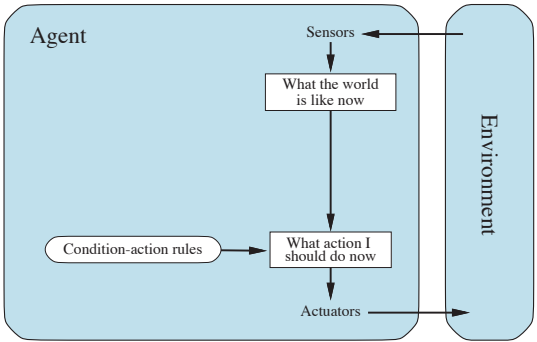
Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partially	Multi	Stochastic	Sequential	Static	Discrete
Backgammon	Fully	Multi	Stochastic	Sequential	Static	Discrete
Taxi Driving	Partially	Multi	Stochastic	Sequential	Dynamic	Continuous
Medical Diagnosis	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Image Analysis	Fully	Single	Deterministic	Episodic	Semi	Continuous
Part-picking robot	Partially	Single	Stochastic	Episodic	Dynamic	Continuous
Refinery controller	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Interactive English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Outline

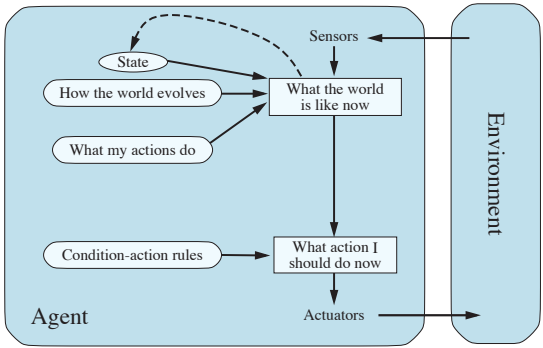
Four basic types in order of increasing generality:

- Simple reflex agents
- Reflex agents with state
- Goal-based agents
- Utility-based agents

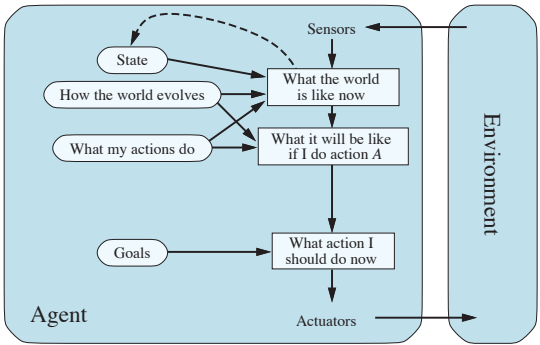
All of them can be turned into learning agents



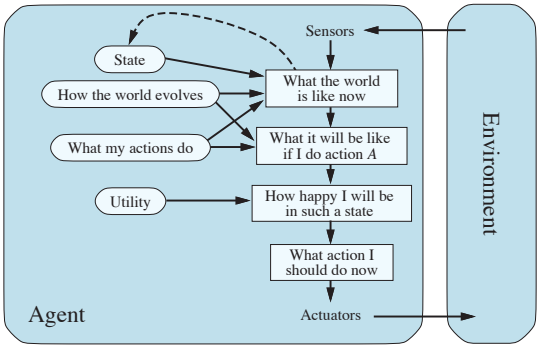
function Simple-Reflex-Agent($percept$)
returns an action
  persistent: $rules$ ▷ *a set of condition-action rules*
  $state \leftarrow$ interpretInput($percept$)
  $rule \leftarrow$ ruleMatch($state$)
  $action \leftarrow rule$.Action
  return $action$



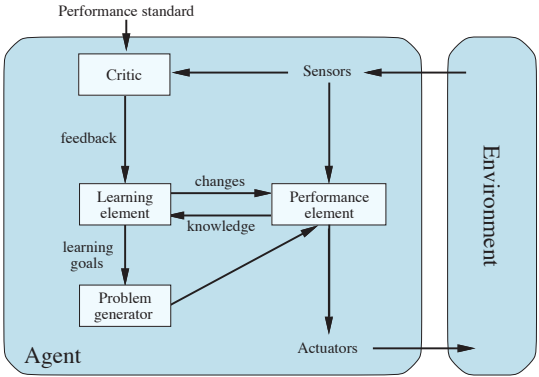
function Model-Based-Reflex-Agent($percept$)
returns an action
  persistent: $state$ ▷ *conception of the world*
  t_m ▷ *env. dynamics model*
  s_m ▷ *sensor dynamics model*
  $rules$ ▷ *a set of condition-action rules*
  $action$ ▷ *the most recent action*
  $state \leftarrow$ updateState($state, action, percept, t\_m, s\_m$)
  $rule \leftarrow$ ruleMatch($state, rules$)
  $action \leftarrow rule$.Action
  return $action$



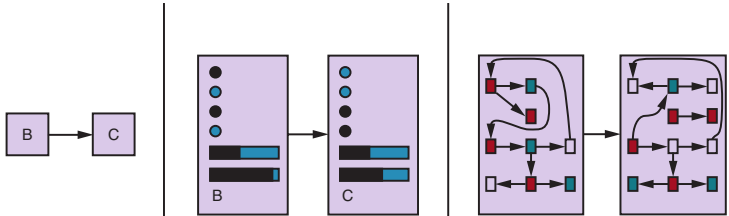
- (Complex) Goals drive behaviour
- Together with a model allows long-term reasoning
- We will see this when we cover the topics of Search (Week 1) and Planning (Week 3)



- Economic notion of (expected) utility drives behaviour
- Helps to deal with uncertainty
- We will see this when we cover the topics of Uncertainty (Week 2) and Stochastic Planning (Week 3)



- Introduces an automatic element to programming agents (learning)
- We will see this in the sessions on Machine Learning (Week 4)



(a) Atomic (b) Factored (c) Structured

- These are high level architectures, algorithms need more detail
- Key aspect of algorithms we will see: state representation
  - Atomic: monolithic states, useful for high-level algorithms
  - Factored: states have discernible internal structure, which we can exploit
  - Structured: internal structure includes relationships between structural elements

- Agents as an abstraction:
  - Intelligent system
  - Operating in an environment
- Environments
  - Define the problem an agent needs to deal with
  - Observability, Determinism, Dynamicity, Discreteness
- Agent architectures
- State representation

Any Questions.