Agents

Agents and environments

The agent function maps percept sequence to actions

\[ f : P^* \rightarrow A \]

An agent can perceive its own actions, but not always its effects.
What is a Software Agent?

• An “intelligent agent” is a persistent computer system capable of flexible autonomous action in some environment.

[Wooldridge & Jennings]
REACTIVE

PROACTIVE
“Flexible”

SOCIAL

Rationality
Rationality ≠ omniscience

A rational agent chooses whichever actions maximize the expected value of the performance measure given the sensed percept sequence to date and prior environment knowledge.
P.E.A.S.

Task Environments

- Performance metric
- Environment
- Actuators (actions)
- Sensors (percepts)
Task Environments

• Performance metric
• Environment
• Actuators (actions)
• Sensors (percepts)
Environment Characteristics

Observability

Full vs. Partial
Deterministic vs. Stochastic

Episodic vs. Sequential
**Ideal Rational Agent**

For each possible percept sequence
Do whatever Action is expected to maximize performance
*(Given information in the percept sequence
and any built-in knowledge)*
Agent Programs

Agent = Architecture + Program
Use a Table??

Function TABLE-DRIVEN_AGENT(percept) returns an action

static: percepts, a sequence initially empty
table, a table of actions, indexed by percept sequence

append percept to the end of percepts
action ← LOOKUP(percepts, table)
return action

Agent Types

Model-Based Reflex

Utility-Based

Simple Reflex

Goal-Based
Simple Reflex

**function** SIMPLE-REFLEX-AGENT(percept) **returns** an action

**static:** rules, a set of condition-action rules

state ← INTERPRET-INPUT(percept)
rule ← RULE-MATCH(state, rule)
action ← RULE-ACTION[rule]
return action

Will only work if the environment is *fully observable* otherwise infinite loops may occur.
Model-Based Reflex

To tackle \textit{partially observable} environments.
- Maintain internal state
- Over time update state using world knowledge
  - How does the world change.
  - How do actions affect world.
  \[ \Rightarrow \text{Model of World} \]

\[\begin{align*}
\text{function} & \quad \text{REFLEX-AGENT-WITH-STATE}(\text{percept}) \quad \text{returns an action} \\
\text{static:} & \quad \text{rules, a set of condition-action rules} \\
& \quad \text{state, a description of the current world state} \\
& \quad \text{action, the most recent action.} \\
\text{state} & \leftarrow \text{UPDATE-STATE}(\text{state, action, percept}) \\
\text{rule} & \leftarrow \text{RULE-MATCH}(\text{state, rule}) \\
\text{action} & \leftarrow \text{RULE-ACTION}[\text{rule}] \\
\text{return} & \text{action}
\end{align*}\]
**Agent types; goal-based**

The agent needs a goal to know which situations are *desirable*.

- Things become difficult when long sequences of actions are required to find the goal.

Typically investigated in **search** and **planning** research.

Major difference: future is taken into account

Is more flexible since knowledge is represented explicitly and can be manipulated.

---

**Agent types; utility-based**

Certain goals can be reached in different ways.

- Some are better, have a higher utility.

Utility function maps a (sequence of) state(s) onto a real number.

Improves on goals:

- Selecting between conflicting goals
- Select appropriately between several goals based on likelihood of success.
Previous agent-programs describe methods for selecting **actions**.

*Where do these programs come from?*

---

**Agent types; learning**

**Performance element:** selecting actions based on percepts.
- Corresponds to the previous agent programs

**Learning element:** introduce improvements in performance element.
- Critic provides feedback on agents performance based on fixed performance standard.

**Problem generator:** suggests actions that will lead to new and informative experiences.
- Exploration vs. exploitation