Intro to Artificial Intelligence Notes

Chapter 1

* Weak AI: machines can act as if they are intelligent
* Strong AI: if it appears to do this, then it is actually thinking (not just simulating thinking)
* Chinese Room: Person inside the windowless room has a infinitely impossibly large book that tell them to write certain things down and send them out the room. The written response makes perfect sense to the response to whatever input was given.
  + “The whole can have qualities that the constitutive parts do not.”
* The four approaches to AI:
  + Acting Humanly: getting computers to do things which people are better at
    - Turing Test: Converse with either a human or a computer, must guess which it is.
  + Thinking Humanly: the automation of activities we associate with human thinking. Intimately tied with cognitive psychology.
  + Thinking Rationally: the “laws of thought” approach
  + Acting Rationally: intelligent behavior in artifacts
    - Rationality: doing the right thing, given what you know
    - Performance measure: some way of evaluating how “good” you are doing
    - Agent: something that acts
    - Rational Agent: one that acts to achieve the best outcome (or best expected outcome).
    - Book is mostly concerted with this approach to AI.

**Agents**

* Two big parts of agents:
  + Sensors: to perceive its environment
  + Actuators: to act upon the environment
* Environment
* Microworlds: small environments with simple agents that lets you focus on specific problems
* Vacuum World: even simpler world
* Percept: agent’s perceptual inputs at any given instant
* Percept sequence: complete history of everything the agent has ever perceived
* Agent function: maps a percept sequence to an action, abstract mathematical description
* Agent program: concrete implementation, running within some physical system
* Rational Agent: (one that does the “right thing”) should select an action that is expected to maximize its performance measure
* Consequentialism: we evaluate an agent’s behavior by its consequences
* Performance measure: how happy we are with the sequence of states
  + As a general rule, it is better to design performance measures according to what one actually wants to be achieved in the environment, rather than according to how one thinks the agent should behave.
* Rationality:
  + Performance measure
  + Percept sequence to date
  + Agent’s prior knowledge
  + Actions the agent can perform
* Rational Agents maximize expected performance, not actual performance
  + For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.
* Agents use information gathering to know what they don’t know by modifying future percepts.
* Omniscience: knowns the actual outcome of its actions and can act accordingly; although impossible in reality
* Information gathering: doing actions in order to modify future percepts
* A priori: ahead of time, not through actual observation
* An agent is autonomous if: it can gather information and learn

The Nature of Environments

* PEAS: Performance, Environment, Actuators, Sensors
* Environment properties:
  + Fully observable: all relevant information is always immediately accessible
  + Partially observable: some relevant information is missing; must be assumed or gathered
  + Unobservable is when the agent has no sensors at all
  + Single agent: everything else in the environment treated as an object
  + Multi-agent: ‘other objects’ will change their behavior based on the actions of our agent, and thus should be considered agents themselves.
    - Can be competitive, cooperative, or both.
  + Deterministic: completely determined by the current state and the action executed by the agent. Nondeterministic is the opposite.
  + Stochastic == nondeterministic
    - Stochastic if it explicitly deals with probabilities
    - Nondeterministic if the possibilities are listed without being quantified
  + Episodic: divided into atomic episodes
    - Next episode does not depend on the actions taken in the previous episodes
  + Sequential: the current decision could affect all future decisions
    - Short-term actions can have long-term consequences
  + Static: if the environment does not change while the agent deliverates.
  + Dynamic: if the environment can change while an agent is deliberating
    - Semidynamic: environment does not change with the passage of time but the agent’s performance score does
  + Discrete: continuous distinction applies to the state of the environment
    - Chess
  + Continuous
    - Taxi driving
  + Known (environment) the outcomes for all actions are given
  + Unknown: the agent will have to learn how it works in order to make good decisions

The Structure of Agents

* The key challenge for AI is to find out how to write programs that, to the extent possible, produce rational behavior from a smallish program rather than from a vast table.
* Agent programs:
  + Simple reflex agents
  + Model-based reflex agents
  + Goal-based agents
  + Utility-based agents
* Simple reflex agent: select actions on the basis of the current percept, ignoring the rest of the percept history
* Model-based reflex agents:

# Solving Problems by Searching

* Informed algorithms
* Uninformed algorithms
* Four-phase problem-solving process:
  + Goal formulation
    - Goals organize behavior by limiting the objectives and hence the actions to be considered
  + Problem formulation
    - Agent devises a description of the states and actions necessary to reach the goal
  + Search
    - Before taking any action, the agent simulates sequences of actions in its model, searching until it finds a sequence of actions that reaches the goal.
    - Each sequence is called a solution.
  + Execution
    - Agent can now execute the actions in the solution, one at a time.
* Open-loop system ignores the percepts which breaks the loop between agent and environment. Used when the model is deterministic.
* Closed-loop approach is better for nondeterministic environments. The system can then monitor the percepts.

Search problems and solutions

* Problem
  + A set of possible states that the environment can be in. This is called the state space.
  + Includes the initial state and goal states.
* We say a set of actions that can be executed are *applicable* in that set
* Action cost function – Action-Cost(s, a, s’)
  + Gives the numeric cost of applying action a in state s to reach state s’

Formulating problems

* Model – an abstract mathematical description
* Abstraction – the process of removing detail from a representation

Problem solving

* Grid world
  + Two-dimensional rectangular array of square cells in which agents can move from cell to cell
* Using states, initial state, actions, transitional model, goal state, action cost

Search Algorithms

* State space describes the (possibly infinite) set of states in the world, and the actions that allow transitions from one state to another.
* Search tree describes paths between these states, reaching towards the goal.
* Best-first search
  + Choose node n with minimum value of some evaluation function f(n)
  + Text

    Description automatically generated with medium confidence

Search data structures

* Node:
  + Node.State: state to which the node corresponds
  + Node.Parent: node in the tree that generated this node
  + Node.Action: action that was applied to the parent’s state to generate this node
  + Node.Path-Cost: total cost of the path from the initial state to this node.
* To store the “frontier” we use a queue:
  + Is-Empty(frontier): returns true only if there are no nodes in the frontier
  + Pop(frontier): removes the top node from the frontier and returns it
  + Top(frontier): returns (but does not remove) the top node of the frontier
  + Add(node, frontier) inserts node into its proper place in the queue
* The three types of queues used in search algos:
  + Priority queue
    - First pops the node with the minimum cost according to some evaluation function, f. Used with best-first search.
  + FIFO queue
    - First-in-first-out queue first pops the node that was added to the queue first
  + LIFO queue
    - Last-in-first-out queue (also known as a stack) pops first the most recently added node

Redundant paths

* Cycles or loopy paths contain repeated states in a search tree.
* Could remember all previously reach states
  + Takes lots of memory
* Could decide to not worry about repeating past states
  + Would end up saving memeory
  + But would waste time examining redundant paths
* Graph search checks for redundant paths
* Tree-like search does not check for redundant paths
* Let’s compromise and check for cycles, but not for redundant paths in general.
  + Our implementation allows for nodes to check chains of parent pointers to see if states appeared earlier in the path.

Measuring problem-solving performance

* Completeness
  + Is the algorithm guaranteed to find a solution when there is one, and to correctly report failure when there is not?
* Cost optimality
  + Does it find a solution with the lowest path cost of all solutions?
* Time complexity
  + How long does it take to find a solution? This can be measured in seconds, or more abstractly by the number of states and actions considered.
* Space complexity
  + How much memory is needed to perform the search?