1 Artificial Intelligence

- An **agent** is an entity that can perceive and act. This course is about designing rational agents.
- Rational behavior: doing the right thing.
- Environment Types: Fully observable; Deterministic; Episodic; Static, Discrete; Single-agent. The counter part: partially observable; stochastic; sequential; dynamic; continuous; multi-agent.
- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators.

2 Problem Solving

- A search problem consists of
  - a state space
  - a successor function (namely **update function** in data mining algorithm series, more namely **recursion** in bullshit technology.)
  - a start state (**initial value**), goal test (**terminating value**) and path cost function (**we say weights in Graph Theory**)
  - Does any one of the above reminds you of **recursion**?
- Problems are often modelled as a state space, a set of states that a problem can be in. The set of states forms a graph where two states are **connected** if there is an operation that can be performed to transform the first state into the second.
- A solution is a sequence of actions (a plan) which transforms the start state to a goal state.
- **State space graph**: A mathematical representation of a search problem.
- **Search Trees**
  - This is a "what-if" tree of plans and outcomes
  - For most problems, we can never actually build the whole tree
- **General Tree Search** Frontier; Expansion; Exploration Strategy.
- **States vs. Nodes** Nodes in state space graphs are problem states; Nodes in search trees are plans. The same problem state may be achieved by multiple search tree nodes.
- **Graph Search** Graph Search still produces a search tree; Graph search is almost always better than tree search.
- DFS graph search needs to store "explored set", which is $O(b^m)$. However, **DFS is optimal** when the search tree is finite, all action costs are identical and all solutions have the same length. However limitating this may sound, there is an important class of problems that satisfies these conditions: the CSPs (constraint satisfaction problems). Maybe all the examples you thought about fall in this (rather common) category.
- The breadth first search and iterative deepening are conceptually and computationally the same. The only difference is the "space" (we call them **memory**) would be partially saved by iterative deepening search.
- **Heuristics** estimate of how close a node is to a goal; Designed for a particular search problem.
- **A star search** Uniform-cost orders by **path cost**, or backward cost $g(n)$; Best-first orders by **distance** to goal, or forward cost $h(n)$. A* Search orders by the sum: $f(n) = g(n) + h(n)$. The distance is an estimated one.
- When A* terminates its search, it has, by definition, found a path whose actual cost is lower than the estimated cost of any path through any node on the frontier. But since those estimates are optimistic, A* can safely ignore those nodes.
- In general, most natural admissible heuristics tend to be consistent, especially if from relaxed problems.
-