Wang Xiyu

# 1 Overview

- Week 1-3: Classical AI, search algorithms
  1. Uninformed search
  2. Local search: hill climbing
  3. Informaed search: $A^*$
  4. Adversarial search Minimax
- Week 4-7: Classical ML
  1. Decision trees
  2. Linear/Logistic regression
  3. Kernels and support vector machines
  4. "Classical" unsuperivese learning
- Week 10-12: Modern ML
  1. Neural networks
  2. Deep learning
  3. Sequential data
- Week 13: Misc.

# 2 AI: Computers Trying to Behave Like Humans

- **PEAS Framework:**
  - **Performance measure:** define “goodness” of a solution
  - **Environment:** define what the agent can and cannot do
  - **Actuators:** outputs
  - **Sensors:** inputs
- Agent function is sufficient.
- Common agent structures (to define an AI agent):
  - Reflex
  - Goal-based
  - Utility-based
  - Learning
  - (Others possible; can mix and match!)
- Exploration vs exploitation

## 3 Problem Statement

fully observable $\wedge$ deterministic $\wedge$ static $\wedge$ discrete $\implies$ only need to observe once
To solve a prob using search:

- A goal or a set of goals
- a model of the enironment
- a search algorithm

goal formulation -> problem formulation -> search -> execute

1. goal formulation
2. problem formulation, eg. path finding
   - states: nodes representation invariant:: abstract states should correspond to concrete states
   - initial state: starting node
   - goal states/test: dest node
     Goal test: define the goal using a function $is\_goal$
   - actions: move along an edge :: $|actions(state)| \leq (branching\_factor)$
   - transition model: $(curr\_state, action) \implies next\_state$
   - action cost function: see edges
3. 

## Search

### Uninformed search

No information that could guide the seaech: mo clue how good a state is

```
create frontir,
...
```

frontier: queue: BFS

### Depth limited search

limit the search to depth l
backtrack when the limit is hit.
time complexity: exponential to search depth
space complexity: size of the frontier

### Iterative deeptening search

seatch with depth from 0 to inf
return soln when found. Both complete

## Concept Proof

## Solution

### Easy