AI in Game Development

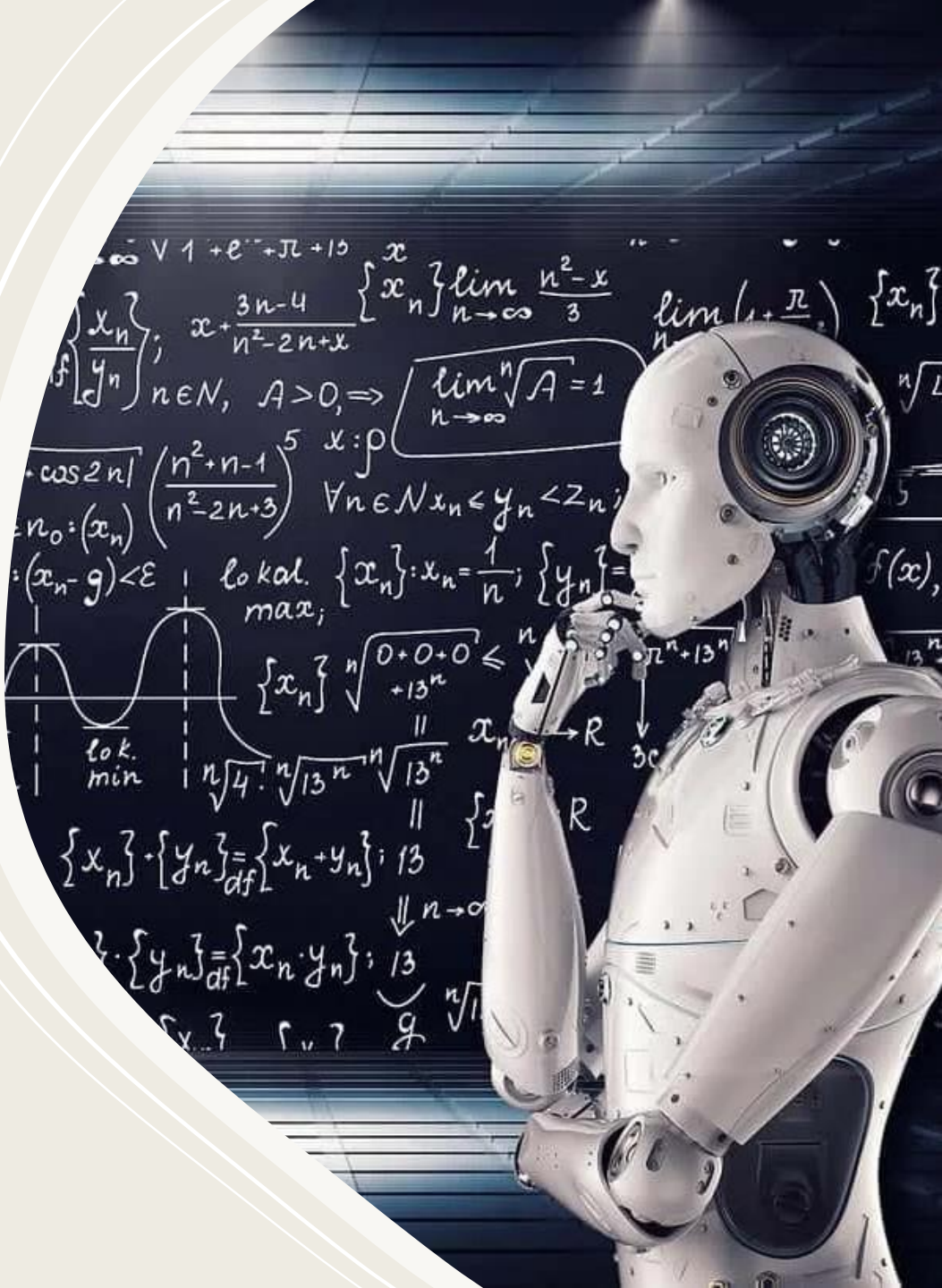
Dr. Mousavi – Shiraz University

By Hossain Khademian – Summer 2021

Definitions

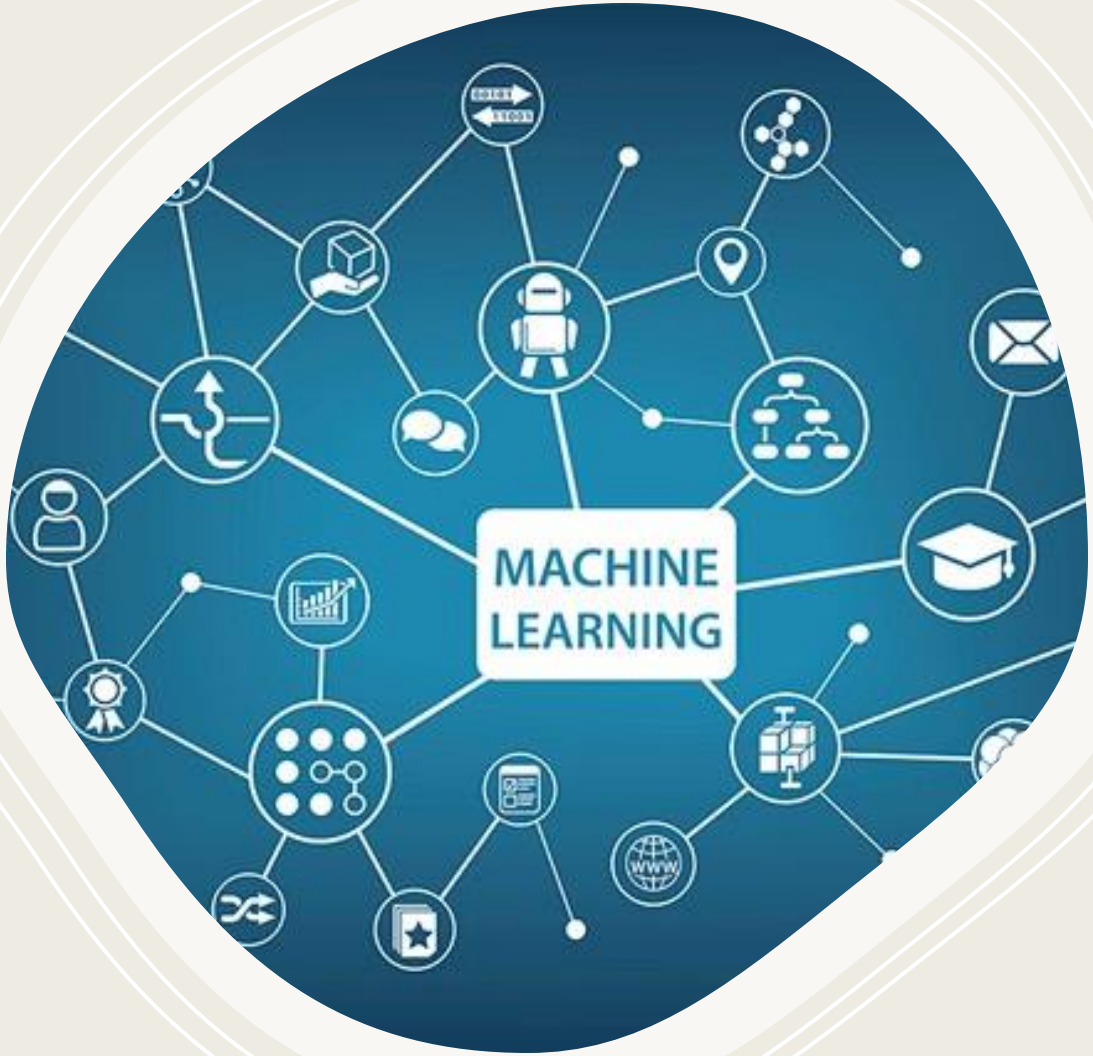
Artificial Intelligence

- As opposed to the Natural Intelligence displayed by humans or animals which have real mind to make decisions.
- AI is intelligence demonstrated by machines.
- There are some general algorithms and problem specific algorithms like Search , CSP , …

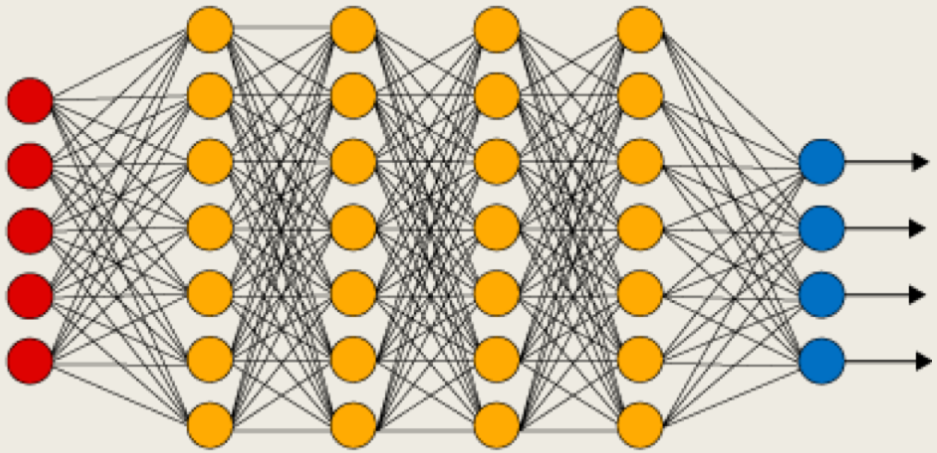


Machine Learning

- Machine learning is a branch of AI
- Focuses on Imitate the way that humans learn
- gradually improving its accuracy
- Heavily use data to learn about problem and improve
- Make predicts and decision upon them based decision tree on what it learned past



See: https://youtu.be/l4Ye70M3rZU



Deep Learning

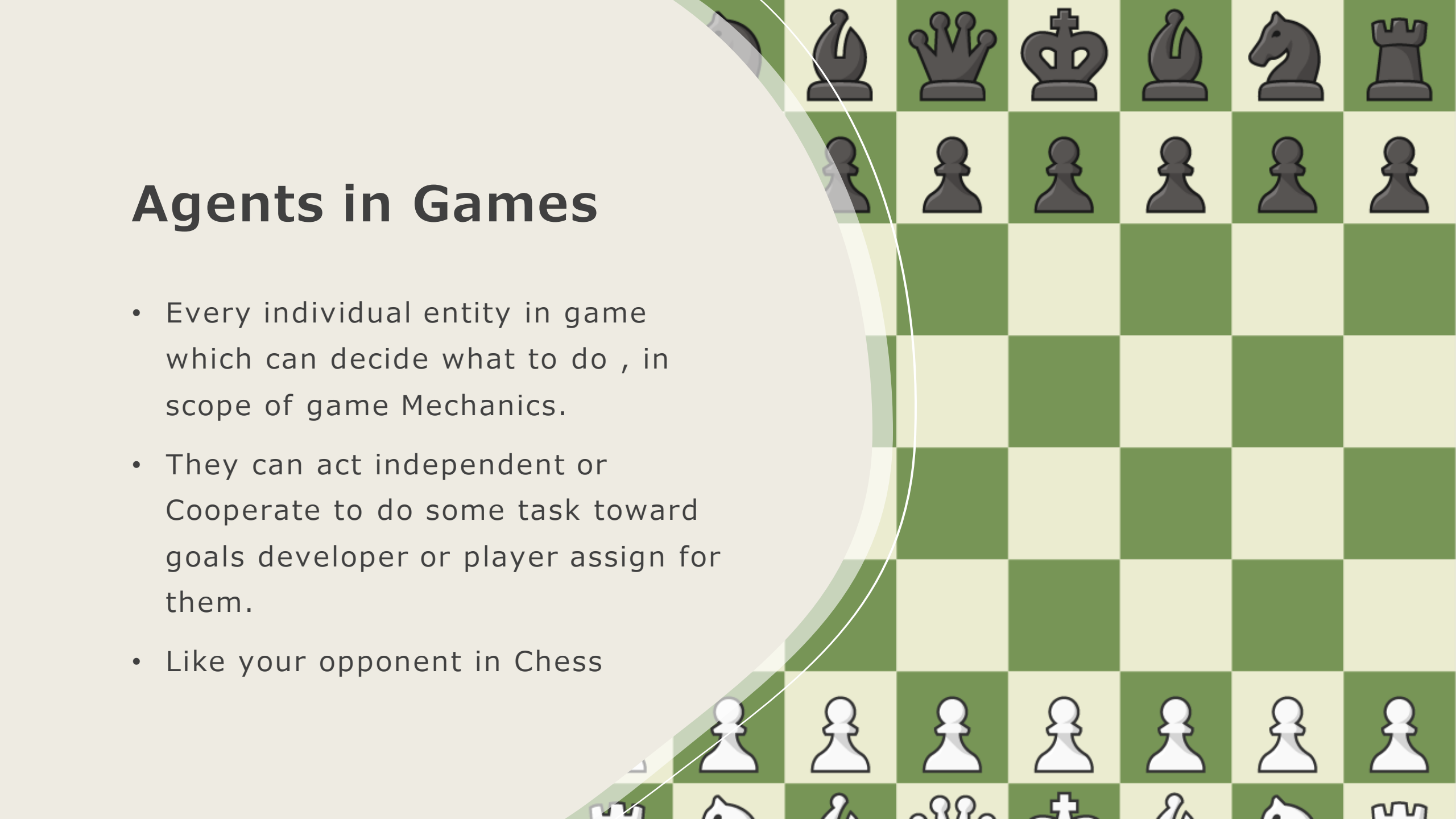
- part of machine learning
- inspired by our brain and the connection between neurons
- Most use neural network architecture
- "deep" in reference to the layers that these neural networks have
- Used in Computer Vision and Speech Recognition

AI for Games

- AI for C.S. is Different for Games
  - Challenging Opponents - Helpful Allies
  - Often constrained by game rules
- Must be smart, but purposely flawed
  - Loose in a fun, challenging way
- **Must perform in real time (CPU)**
- Configurable by designers
  - Not hard coded by programmer

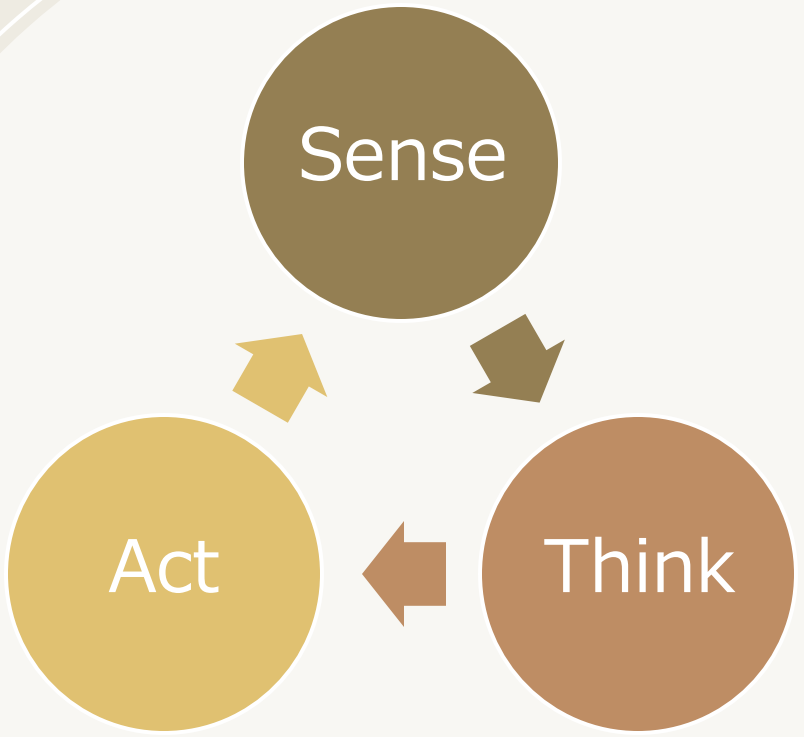
Agents in Games

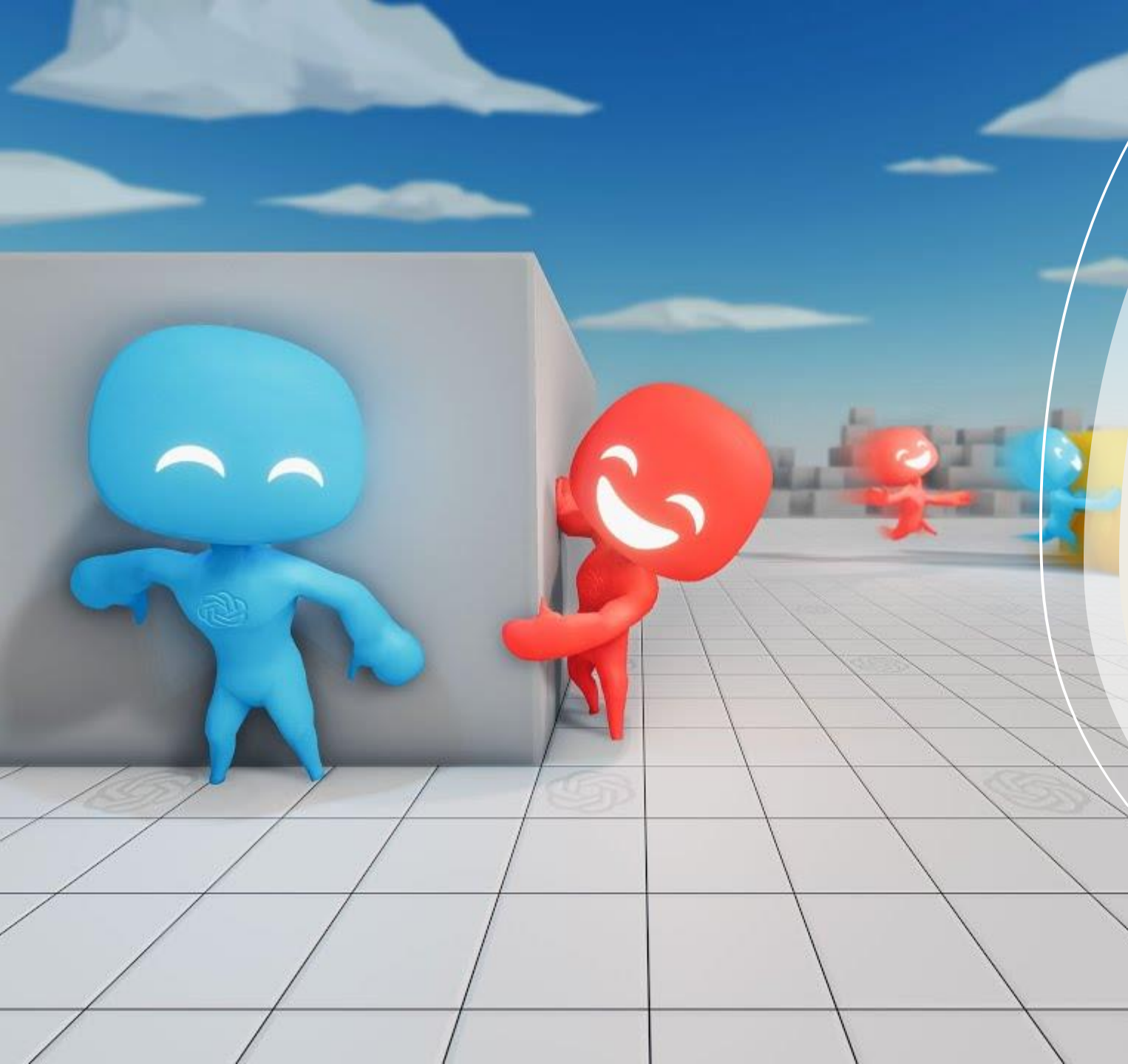
- Every individual entity in game which can decide what to do , in scope of game Mechanics.
- They can act independent or Cooperate to do some task toward goals developer or player assign for them.
- Like your opponent in Chess



Agents

- Most AI focuses around game agent
  - think of agent as **NPC**, enemy, ally or neutral
- Loops through: sense , think, act cycle
- Acting is **event specific**:
  - first sense and think then act





Game Agents Sensing

- Gather current world state: barriers, opponents, objects
- Needs limitations: **avoid cheating** by looking at game data
  - Typically, same constraints as player (vision, hearing range, etc.)

Sense - Vision

- Quite complicated to test visibility
- Compute vector to each object

1. magnitude (is it too far away?)
2. Check angle (dot product) (within 120° viewing angle?)
3. Check if obscured (Most expensive: so do last)

- https://youtu.be/3-jPo2wzvdw

Sense

Hearing

- Example:
  - tip-toe past, enemy doesn’t hear
  - run past, enemy hears
- Implement as event-driven
  - *When player performs action*
  - notify agents within range

Communication

- Sensing data from other agents
  - instant (connected by radio)
  - hearing (loud shout)

Reaction times

- Sensing may take some time
- Build in delay Implement (simple timers)

Game Agents Thinking

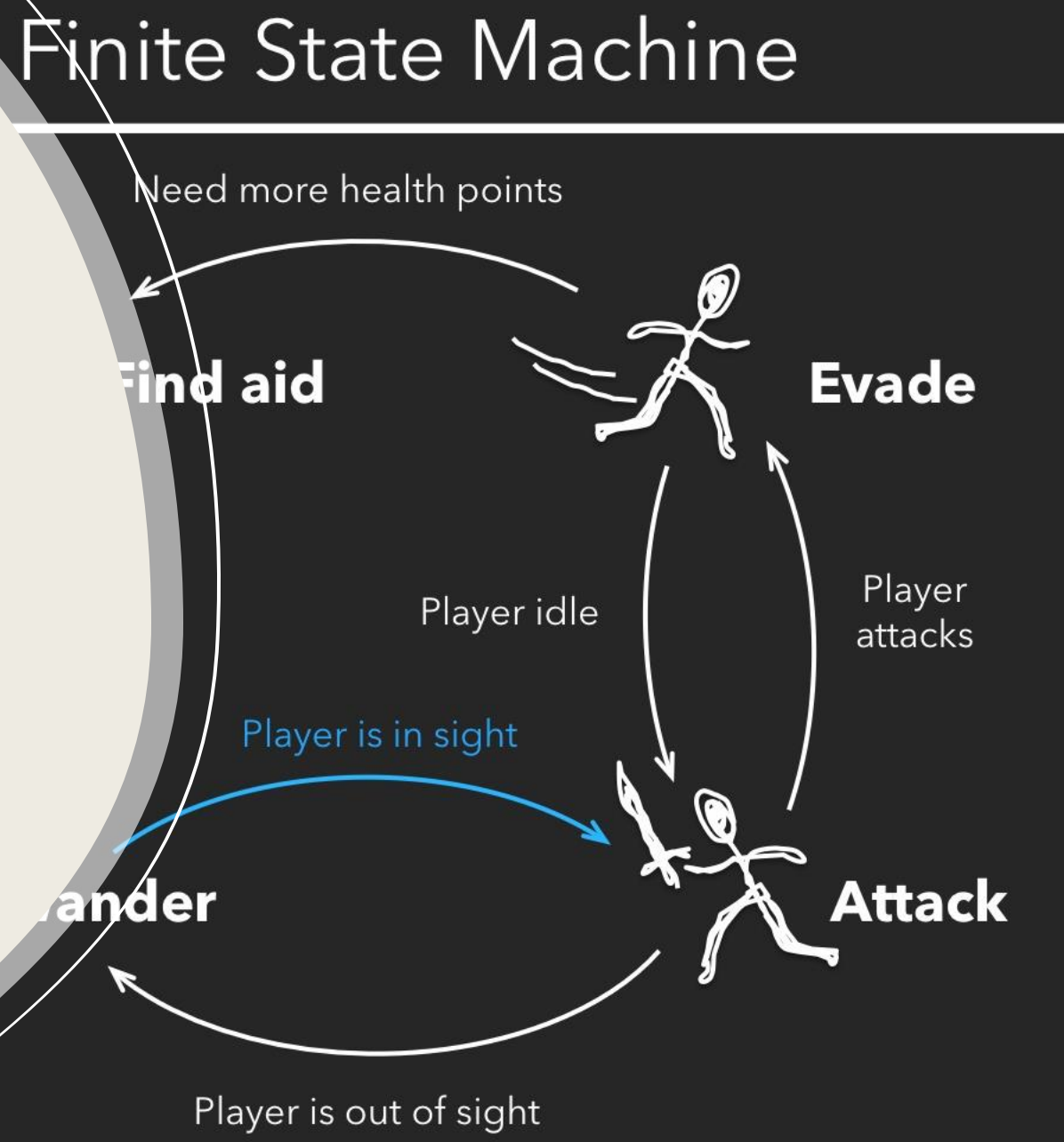
- Evaluate information
- Make decision

1. Pre-coded expert knowledge
   - Typically hand-crafted “if-then” rules
   - + “randomness” to make unpredictable
2. Search algorithm for best (optimal) solution (MinMax, MinConflict, ...)

Thinking

Finite State Machines

- FSM *most popular* . Appealing since:
  - simple, natural, embodies common sense and knowledge of domain
  - Ex: See enemy weaker than you? -> Attack. See enemy stronger? -> Go get help
- But does not scale
  - Complex situations have many factors
  - Add more rules, becomes brittle
- Suitable for many AI tasks:
  - Many agents have quite narrow domain



See: https://www.kidscodecs.com/finite-state-machine-games/

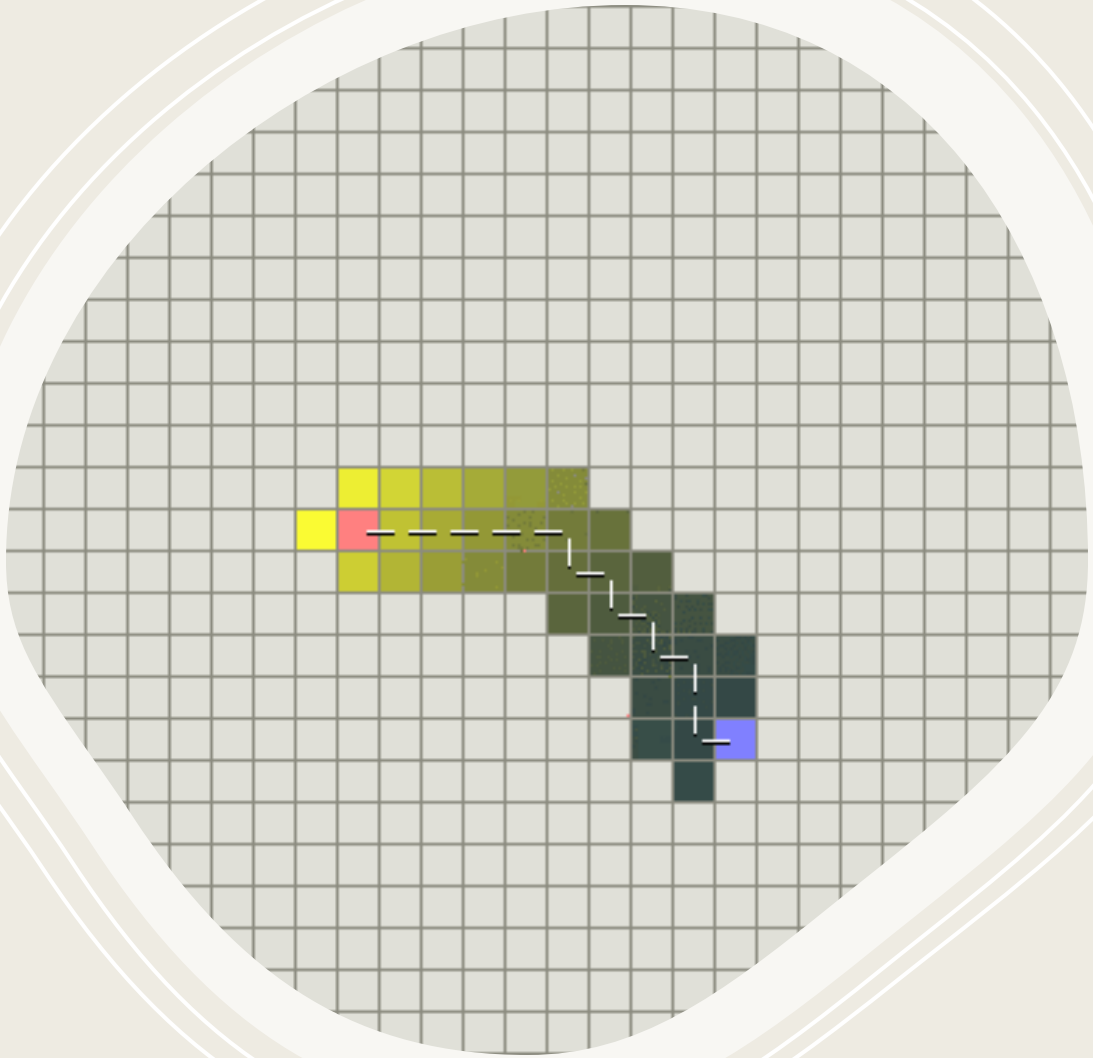
Thinking

Search

- Look ahead and see what move to do next
  - Ex: piece on game board (MinMax , MinConflict, ...) , pathfinding (A* , hueristic)

- Works well with known information
  - can see obstacles, pieces on board, path finding, …

  - Collect Knoledge about environment and query when needed (Resolution , …)

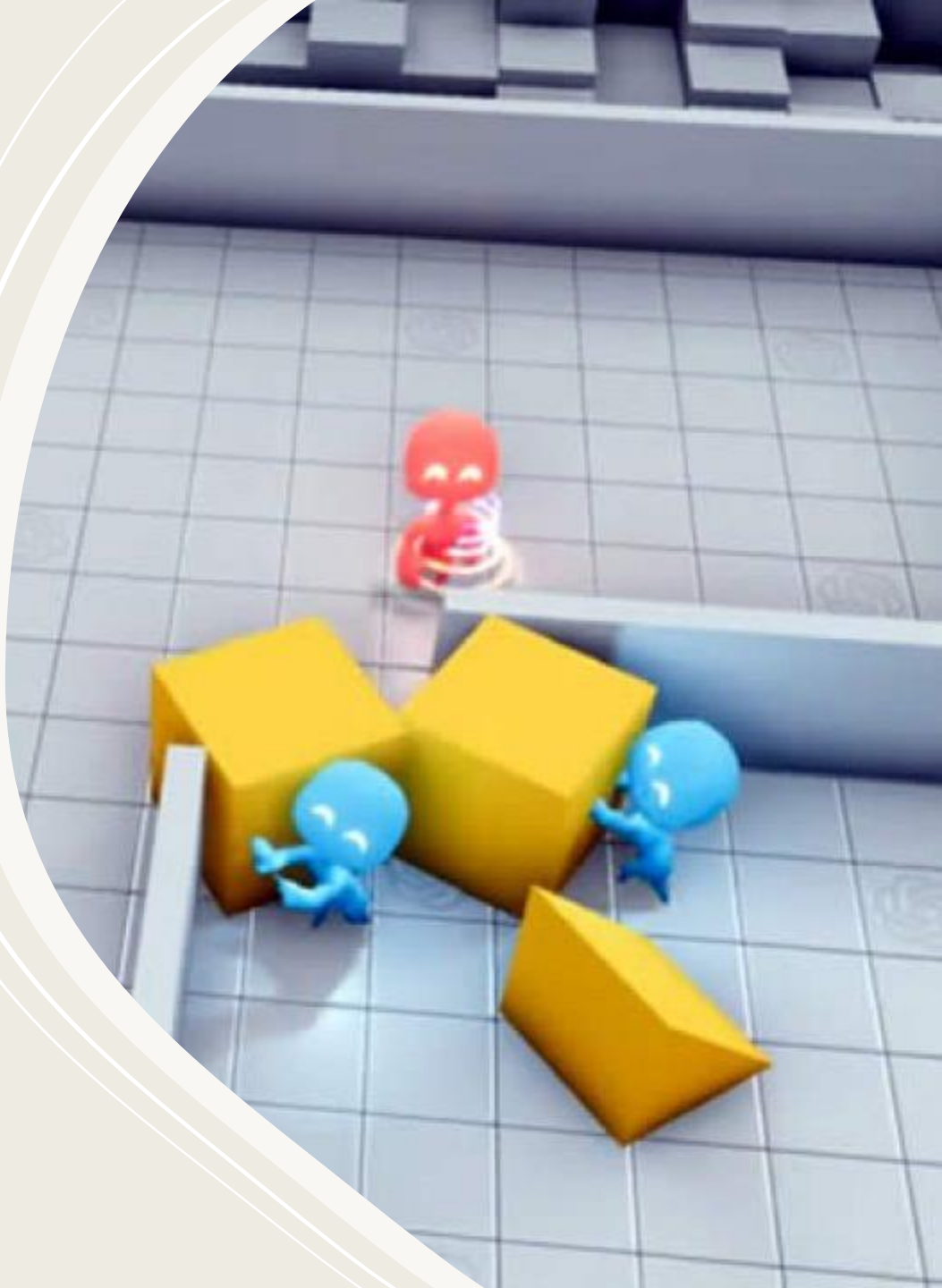


Thinking
Machine learning

- Evaluate past actions, use for future action
- Even we can train Agents in Game Development and use in Production as Mutate Agent
- Learning process is too slow and costly
- Required large investments in development to buy equipments and hire technitions to run tests and players to learn from

Game Agents – Acting

- Learning and Remembering
- Not important in agent short-lived (enemy drone)
- Helpful if alive for 30+ seconds
  - (player attacks from right, so shield right)
- Implementation, too avoid too much information:
  - fade memory (time , overflow)



Game Agents **Acting**

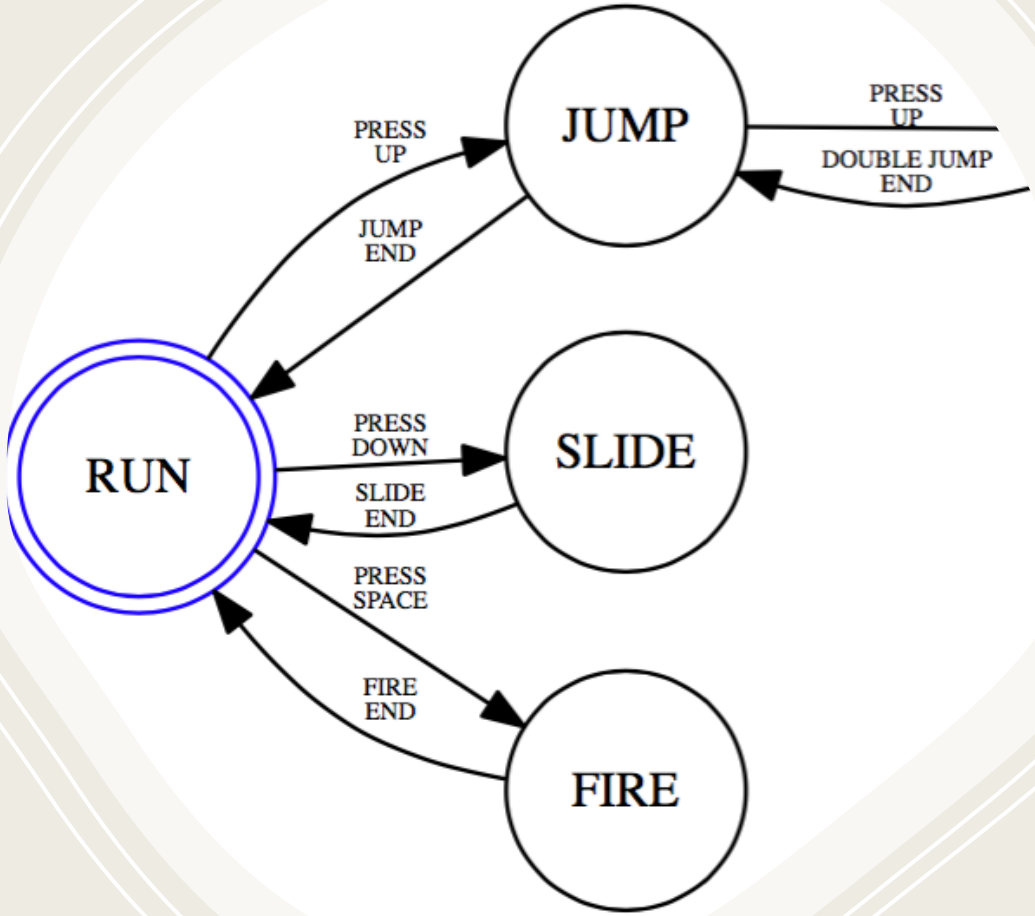
- Making agents stupid
  - Many cases, easy to make agents dominate
  - FPS bot always makes head-shot
- Dumb down by giving “human” conditions
  - longer reaction times, make unnecessarily vulnerable, have make mistakes
- Agent cheating
  - Ideally, don’t have unfair advantage
  - (such as more attributes or more knowledge)
  - may cheat to make a challenge

Algorithms

General and Problem Specific Solutions

Finite State Machines

- Abstract model of computation
  - Set of states
  - A starting state
  - An input vocabulary
  - A transition function that maps inputs and the current state to a next state
- ResState := trans(CurState, inp)



C.S. Search Algorithms

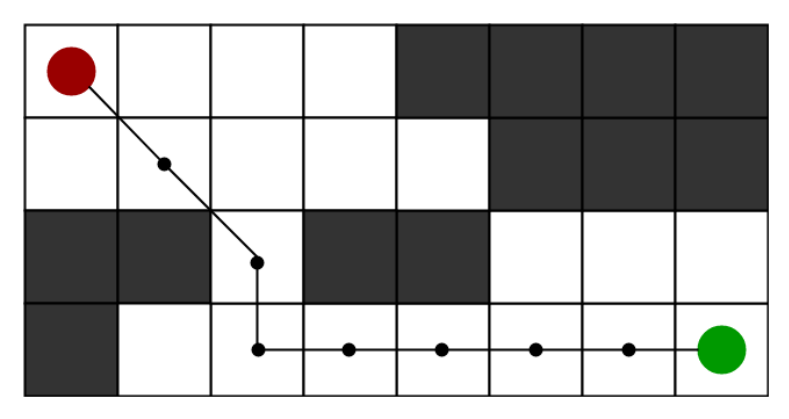
- Convert problem states to a graph
- Current State is start node
- Each action is an edge from a node to other
- Problem Goal (which we want to reach) is Graph's final State
- There is two kind:

1. Blind (UnInformed about next state) :

- DFS, BFS, Iterative-Deeping

2. Informed (can guess what is in the next state)

- Heuristic , A* , ⋯
- SEE:
- https://qiao.github.io/PathFinding.js/visual/



MiniMax

- Heart of board game
- Applies to games where:

1. Players take turns
2. Have perfect information

– *Chess, Checkers, Tactics*

- can work with chance or without perfect information:

- Poker, Monopoly, Dice

Top References

1. https://www.ibm.com/cloud/learn/machine-learning
2. https://blog.bismart.com/en/difference-between-machine-learning-deep-learning
3. https://www.zdnet.com/article/what-is-ai-everything-you-need-to-know-about-artificial-intelligence/
4. https://en.wikipedia.org/wiki/Artificial_intelligence