Artificial Intelligence in Games

CS3005
Digital Media & Games

Hamid Arabnejad
Alireza Jahani

Brunel University London
Monday, 18 October 2021

Contents: 1st half

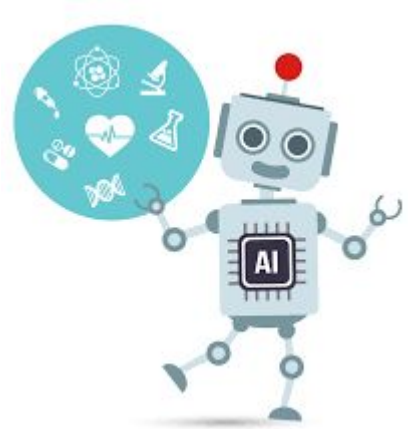
- What is artificial intelligence?
- Game AI vs. academic AI
- What is Game AI?
- Execution Flow of an AI Engine
- History of AI.
- Decision Making Approaches:
    - Finite-State Machines (FSMs)

Contents: 2st half

- Real-time AI
- Machine Learning (ML)
    - Clustering
    - Decision trees.
        - Max-Min
        - Monte Carlo Search Tree
    - AI pathfinding
    - Artificial Neural Networks (ANN).
    - Reinforcement Learning (RL)
    - Deep Reinforcement Learning (DRL)

What is Artificial Intelligence?

Artificial Intelligence is the development of computer systems that are able to perform tasks that would require human intelligence.



Examples of these tasks are visual perception, speech recognition, decision-making, and translation between languages.

Real Life A.I. Examples

- Self Driving Cars
- Boston Dynamics
- Navigation Systems
- ASIMO (humanoid robot created by Honda)
- Chatbots
- Human vs Computer Games
- Many More!

Dream

“By the middle of the 21st century, a team of fully autonomous humanoid robot soccer players shall win a soccer game, complying with the official rules of FIFA, against the winner of the most recent World Cup.“

https://www.robocup.org/objective

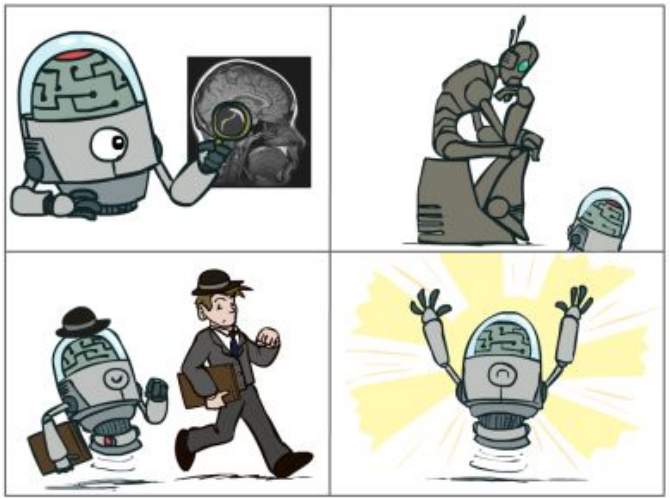
- Why football?
  - Who do you think will win that match?
    - Does is it really matter who wins?

AI is NOT just building Robots!

Robots are “**real object representation of an AI engine**”

An AI Engine is considered something that can do one or more of the following

Think like people



Think rationally

Act like people

Act rationally

Are humans rational?

Humans are not entirely rational, but they can think and behave rationally or not, depending on whether they apply, explicitly or implicitly, the strategy of theoretical and practical rationality to the thoughts they accept and to the actions they perform.

While, within Artificial Intelligence, a *rational agent* is typically one that **maximizes its expected utility**, given its current knowledge.

https://plato.stanford.edu/entries/logic-ai/

Game AI vs Academic AI

Game AI

1) Focus on creating convincing outputs.
2) Strong reliance on “rules of thumb”, or heuristics.
3) Cheating can be used to help solve AI shortcoming.
4) Wide range of goals.
   E.g., overly strong AIs can be bad.

Research AI

1) Focus on creating intelligent software.
2) Focus on:
   a) machine learning
   b) ability to perform general intelligent actions
   c) ability to handle arbitrary data.
3) Cheating isn’t possible in robotics.
4) Aim for strongest possible AI.

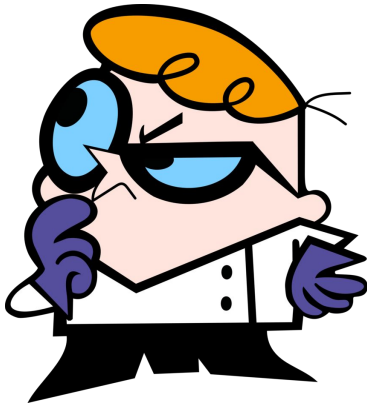
What is Game AI?

- Many think that game AI is just there to be the enemy
- that may have been true in the past, but now AI does much more than that
- AI tries to imitate believable behavior (Like humans) in *Non-Player Characters (NPCs)*
- Make the game “feel” real.
- Obey laws of the game
- Show decision making and planning

Purpose of AI in games

Non-Player Characters (NPCs)

What makes a good NPC?



What makes a good NPC?

01	Right level of challenge	• Too stupid and the player will feel bored. • Too skilled and the player will feel cheated.
02	Right level of predictability	• Too predictable will cause players to “game” the NPCs. • Too unpredictable may damage the sense of realism / immersion.
03	Variety	• Even Pac-Man came with four different AIs. • NPCs types need to stand out from each other.

The goals of Game AI are:

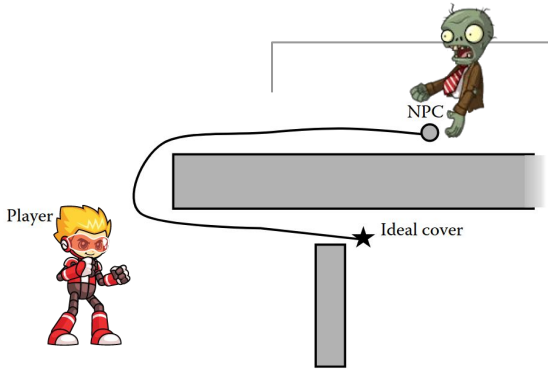
- Intelligence!=Fun
  - be fun, make the player come back
- Be challenging but not overwhelming (unless the hardest difficulty is selected)
- Make sure the AI does not cheat! (At least do not get caught 😜)
  - different than the AI getting bonuses with difficulty changes such as damage, speed
- on a more technical side, Do not hog all the resources! 😟 (CPU time)
  - on average, AI gets 30% of the CPU time

Endure and Survive: the AI of The Last of Us

every character has two sets of behavioral traits

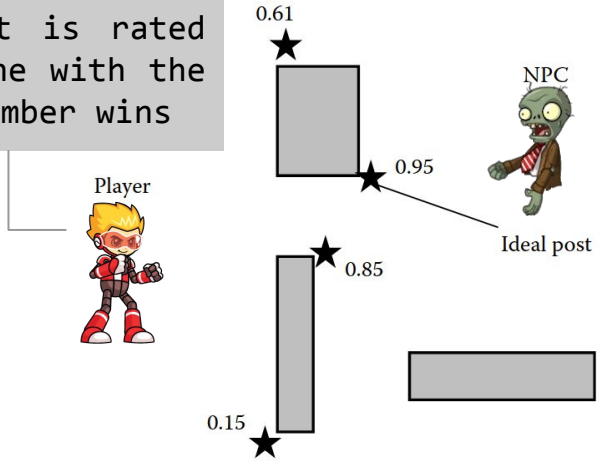
- on the offensive front (opponent AI side)
- on the defensive side(player's side).

It all depends on how the players respond to the situation which involves different characters. The AI switches based on these responses and acts accordingly



If the path to the cover required running toward the player for too long, as in this example then the cover would be rejected

Every post is rated and the one with the highest number wins



Game AI : Observations

AI has three basic game roles

- Replacement for human opponents and players
- Support characters for interesting player interaction
- Units for player management

Entertainment is much more important than realism

- Cheating is OK if user can't detect it
- Play to lose or at least make it challenging
- Must include variable levels of skills

NO single type of AI is right for all games or all AI roles

https://web.eecs.umich.edu/~sugih/courses/eecs494/fall06/lectures/lecture13-gameai.pdf

Execution Flow of an AI Engine

Game

Sense

What should be sensed?

Think

Act

Decision Making

- Finite-State Machines
- Decision Trees
- Rule-Based Systems
- Neural Networks
- Fuzzy Logic
- Movement and pathfinding

Execution Flow of an AI Engine

Core concepts: search

Game

Sense

What should be sensed?

Think

Decision Making

- Finite-State Machines
- Decision Trees
- Rule-Based Systems
- Neural Networks
- Fuzzy Logic
- **Movement and pathfinding**

Act

History of AI

Pac-Man

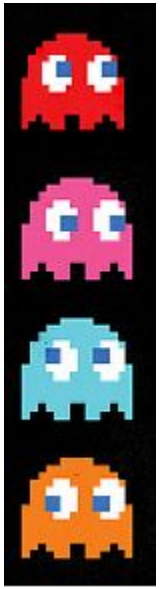
- Developed by Namco
- Distributed by Midway Games (in US)
- Released: 1980
- Single player only
- Simple maze game

History of AI

Pac-Man Ghosts

Ghosts move in a random order and each of them has their own algorithm to control their actions.

- Chase Pac-Man
- Try to get in front of Pac-Man
- Move at Random



History of AI

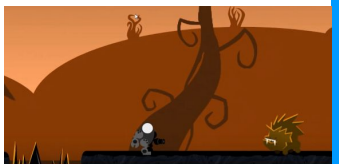
Super Mario Bros.

- Developed by Nintendo
- Released in 1985
- Classic example of the Platformer

History of AI

Super Mario - Scripted AI

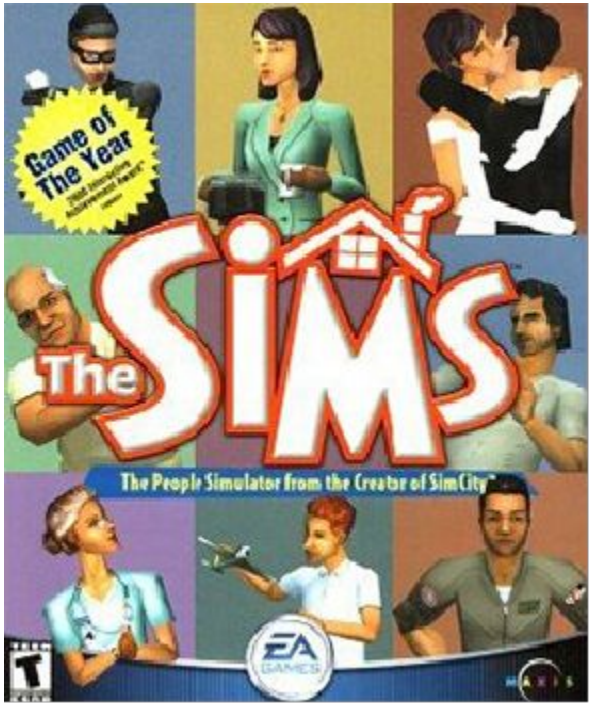
- Enemy units in the game are designed to follow a scripted pattern.
- Either move back and forth in a given location or attack a player if nearby
- Became a staple technique for AI design



History of AI

The Sims

- Developed by Maxis
- Published by Electronic Arts
- Released in 2000
- A life simulation game in which the player has control over a semi-autonomous people

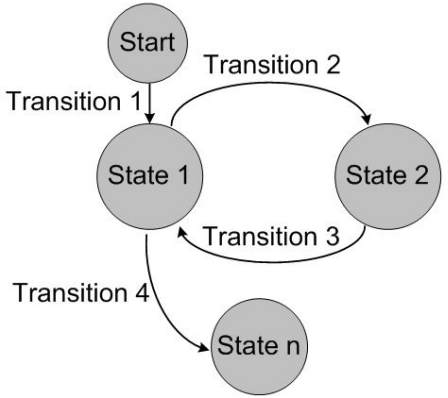


History of AI

The Sims - Complex characters

- Full-blown agent-based simulation.
- Dynamically models emotional interactions and relationships between characters.
- Characters also have defined needs and personality traits that dictate how they act in given situations.

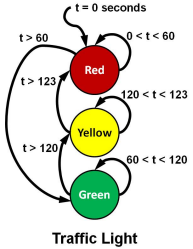
Finite-State Machines (FSMs)



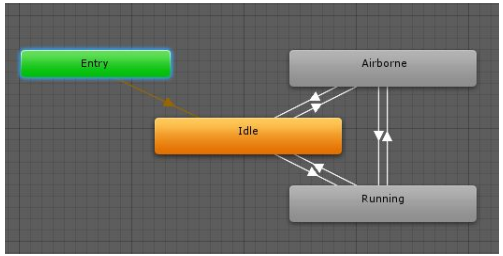
Simple theoretical construct

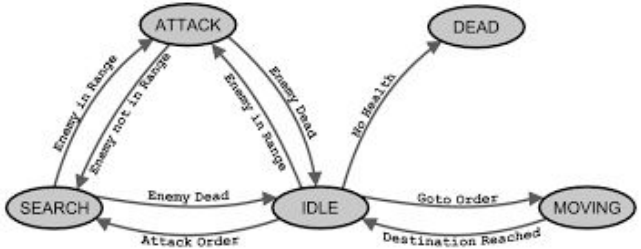
- Set of states (S)
- Input vocabulary (I)
- Transitional function T(s,i)

A way of denoting how an object can change its state over time.



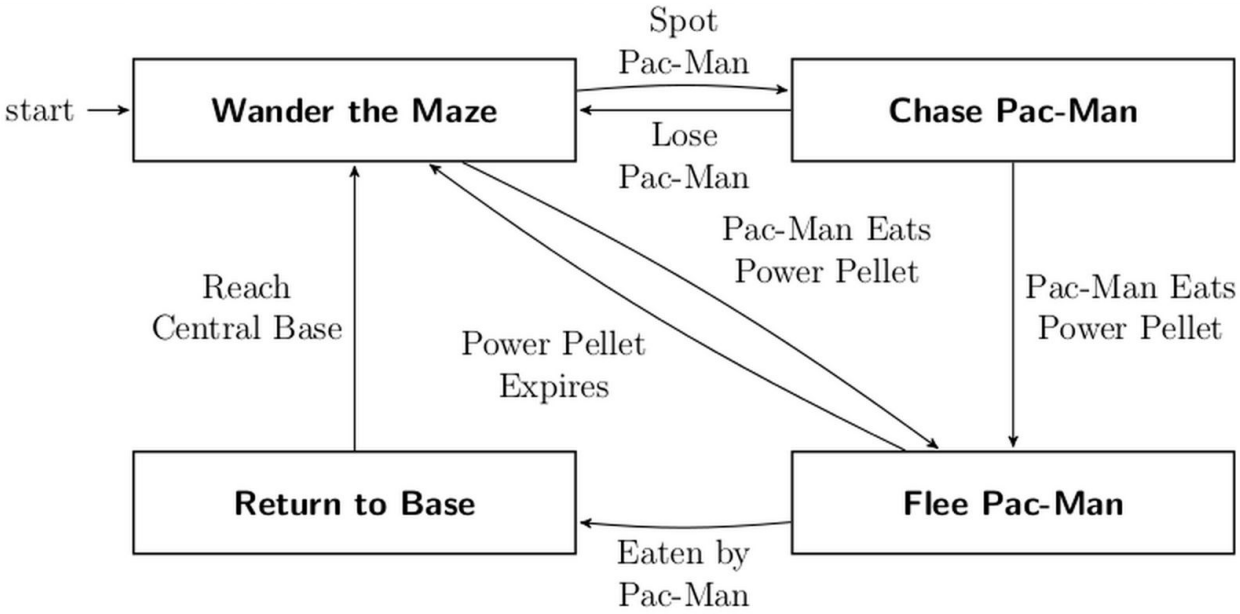
Traffic Light





Finite-State Machines (FSMs)

Example : Pac Man



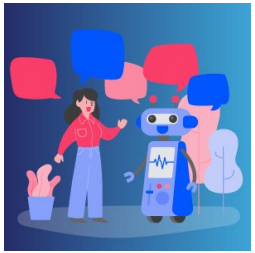
16 Games With Incredible Artificial Intelligence

Real-time AI

Criminal Investigation

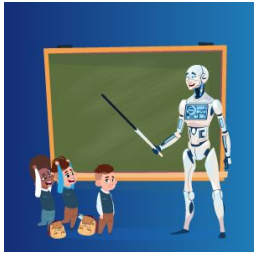
Criminals identification

Natural Language Processing



humans to interact with the computer

Education



personalized learning

Banking

assists in detecting credit card frauds

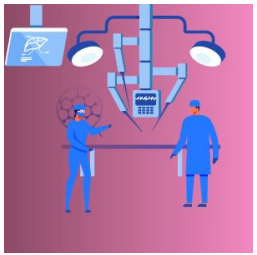
Entertainment

Recommender systems like Netflix

Speech Recognition

understand many languages, accents, and slangs

Healthcare

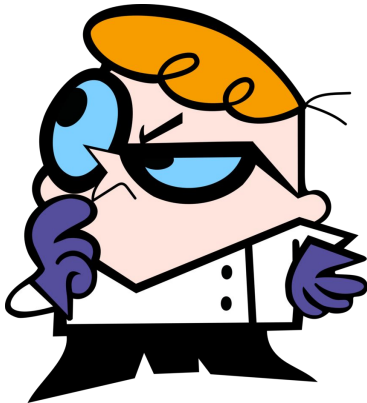


analyzing the patient data

source

Real-time AI

what does one need for a Real-time AI?



Real-time AI

what does one need for a Real-time AI?

- Observe environment in real-time.
- Find a favourable location.
- Move using best route.
- Choose a favourable activity.
- **Do all this in real time(!)**

Game: Half-Life 2

Classification, pathfinding, decision making, good performance.

Game AI vs Academic AI

Game AI

1) Focus on creating convincing outputs.
2) Strong reliance on “rules of thumb”, or heuristics.
3) Cheating can be used to help solve AI shortcoming.
4) Wide range of goals.
   E.g., overly strong AIs can be bad.

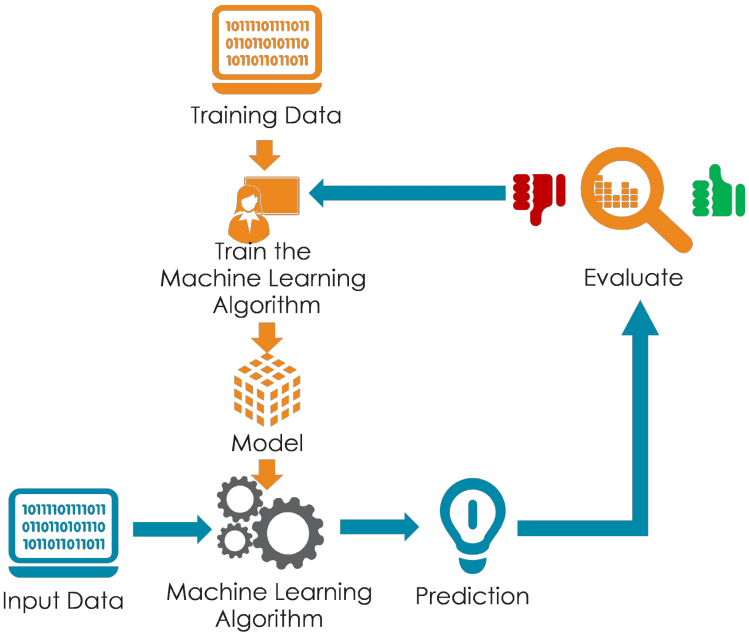
Research AI

1) Focus on creating intelligent software.
2) Focus on:
   a) machine learning
   b) ability to perform general intelligent actions
   c) ability to handle arbitrary data.
3) Cheating isn’t possible in robotics.
4) Aim for strongest possible AI.

Machine Learning (ML)

Why

1) Machine learning enables programs to learn without being explicitly programmed.
2) Essential element for self-learning AIs.
3) Allows algorithms that can incorporate gameplay data

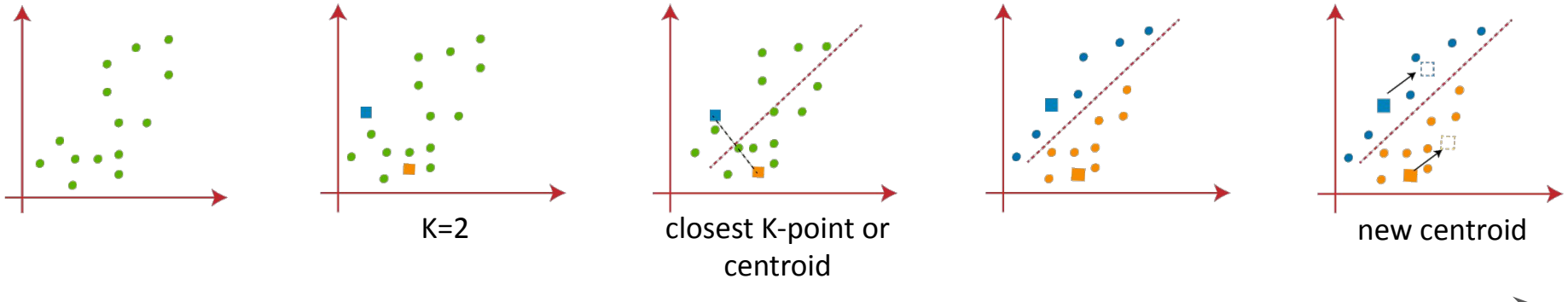


Machine Learning (ML)

How

1) Obtain collection of relevant data, e.g.:
    a) Outcomes, when a certain move was chosen in past instances.
    b) Output of random simulations of the game.
2) Reduce the complexity of the data,
    a) to make analysis tractable.
3) Cluster & categorize the data.
    a) To find strong correlations between the selected move and the outcome.
4) Establish a weighting algorithm which favours move with a more likely favourable outcome.

Machine Learning Techniques : Clustering



Machine Learning Techniques : Clustering

let's say you are dealing with a team-based First Person Shooter (FPS) game:

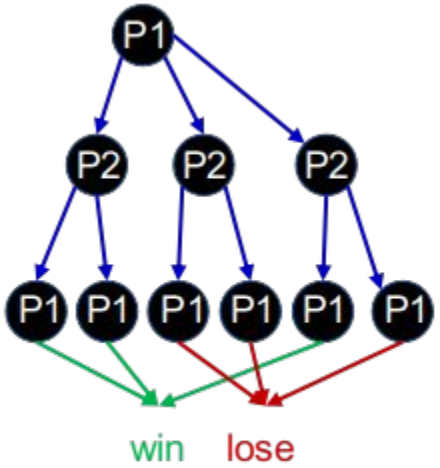
- target : investigate how the game is being played

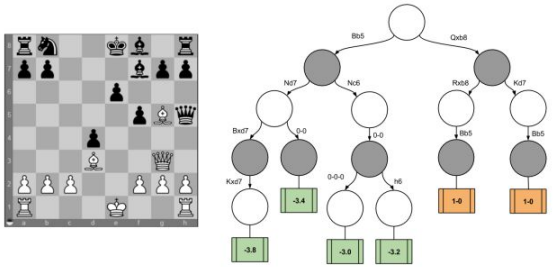
1) collect metrics on:
   a) kill/death ratios
   b) time spent in particular modes (on foot, driving, being a passenger etc.)
   c) various other event data
2) Running a cluster analysis :
   a) you find several different clusters, but note that the majority of the players belong to one of the clusters
3) Investigating this cluster in more detail :
   a) you find that these are players who spend all of their time driving vehicles, and very little time on foot
4) Conclusion :
   a) It would be reasonable to suspect that this group of players love vehicle based combat

source

Decision Trees

- A schematic overview of decision moments in the game:
  - Circles are states.
  - Arrows are choices.
- Can apply for both player and AI decisions.
- Each possible decision can be scored by its expected outcome, allowing algorithms to rank decisions.

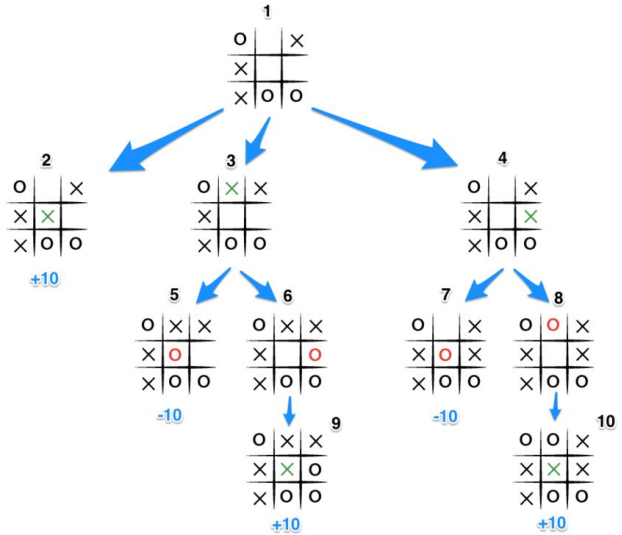




Game decision tree: differences in complexity

Tic-Tac-Toe

- Limited options per turn.
- Deterministic outcomes.



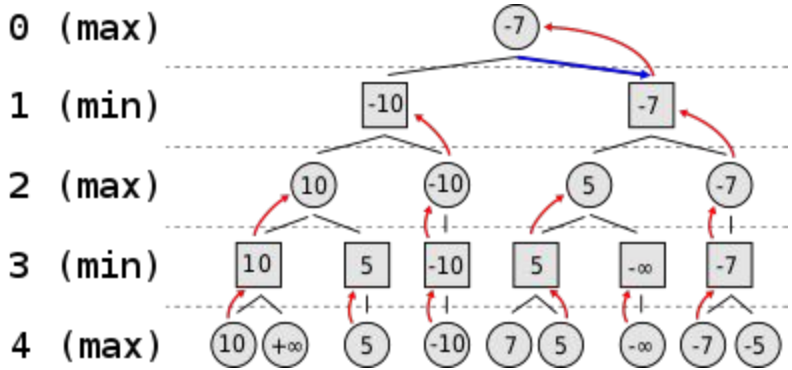
Tactical RPG (Role Playing Games)

- Many options per turn…
- Random factors, full tree hard to analyze.

Decision trees : Max-Min

➔ A strategy defines a complete plan of action for a given player.

➔ Given enough processing time an **optimal strategy** can be found for games of **perfect information** by enumerating paths of a game tree.



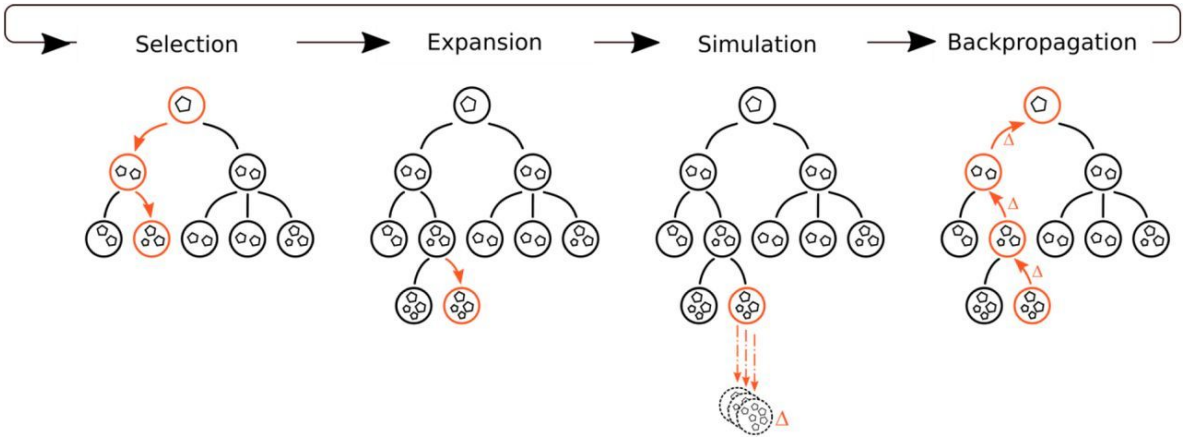
However, in practice this can ONLY be done for small games.

Monte Carlo Search Tree - MCST

MCST embodies the strategy of using random trials to solve a problem.

It is *impossible* to consider all of possible moves in chess.

- So in these games the MCST would randomly choose some of the possible moves to start with. Therefore, outcomes become much more uncertain to human players



AI pathfinding

Pathfinding algorithms

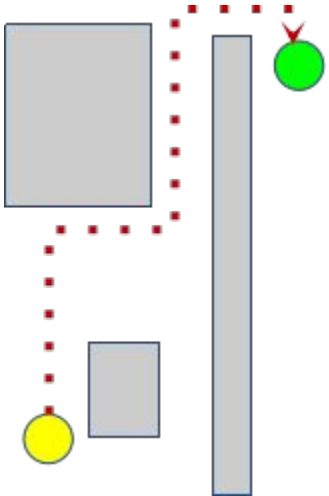
Generate a path to get from A to B, avoiding obstacles.

- Redone for every move action.

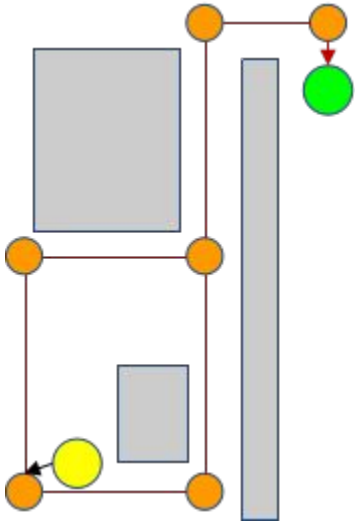
Navigation meshes

Fixed meshes with straight interconnections.

- Short-distance pathfinding may still be needed.

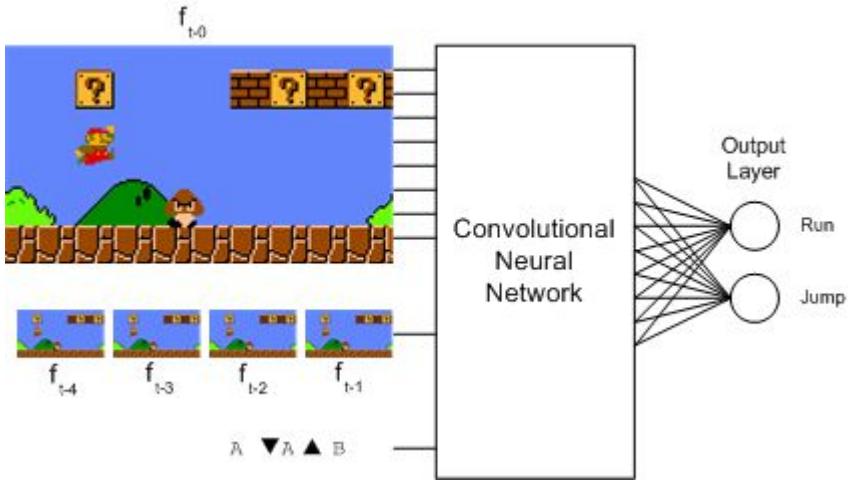


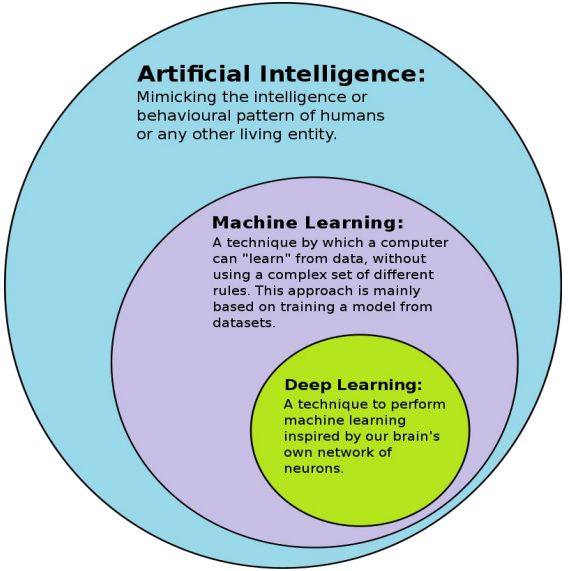
For more examples, please read the Appendix



Deep Learning (DL)

Deep Learning is a subset of machine learning which is essentially a neural network with three or more layers.

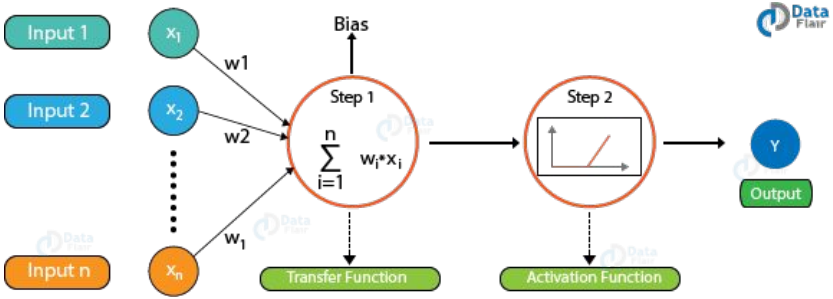


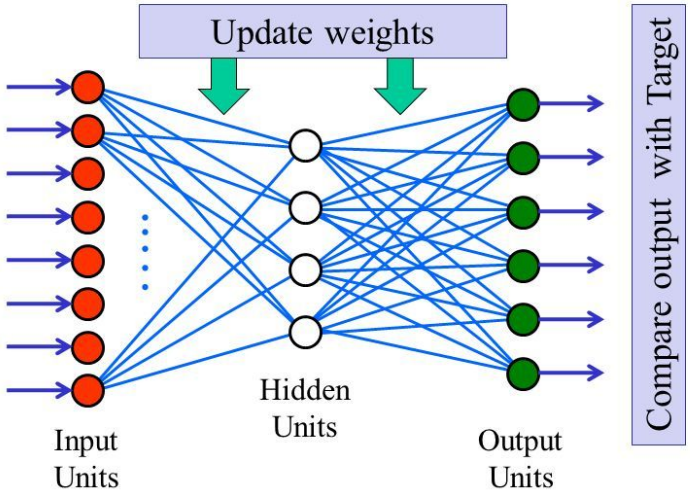


Artificial Neural Networks (ANN)

ANN is a computational model very similar to the structure of our brain.

- Input layer, processing (hidden) layer, output layer
- Each of the connections has a number associated with it called the connection weight and each of the neurons has a number and a special formula associated with them called a threshold value and an activation function respectively
- Learns by example

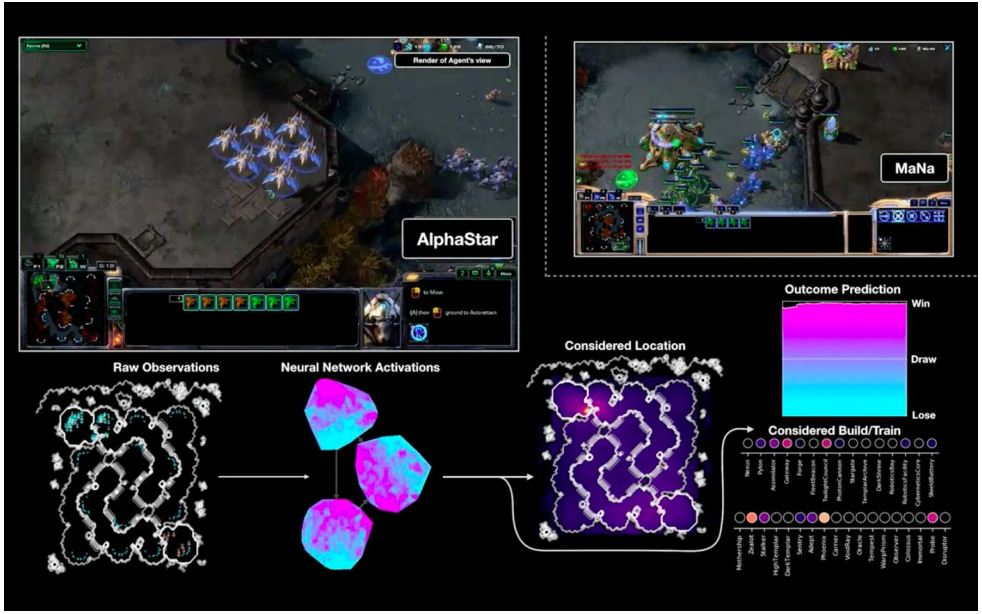




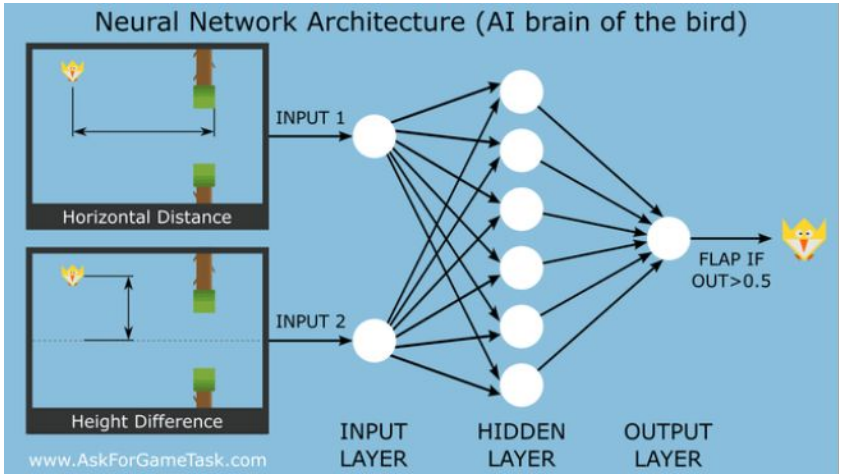
How to build a simple neural network in Python

Artificial Neural Networks (ANN)

A graphical representation of AlphaStar's processing. The system sees whole map from the top down and predicts what behavior will lead to victory.



[Flappy Bird using Neural Network (Github)](#) ([Demo](#))

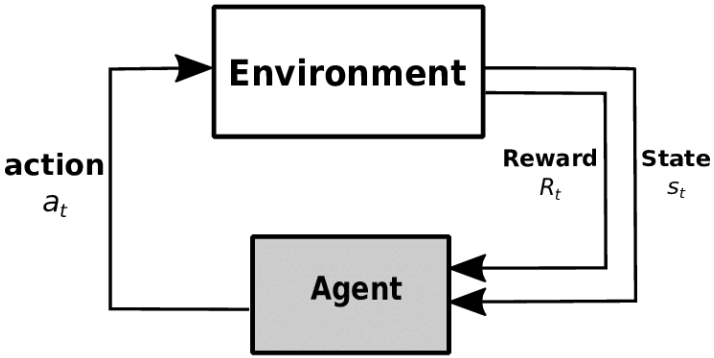


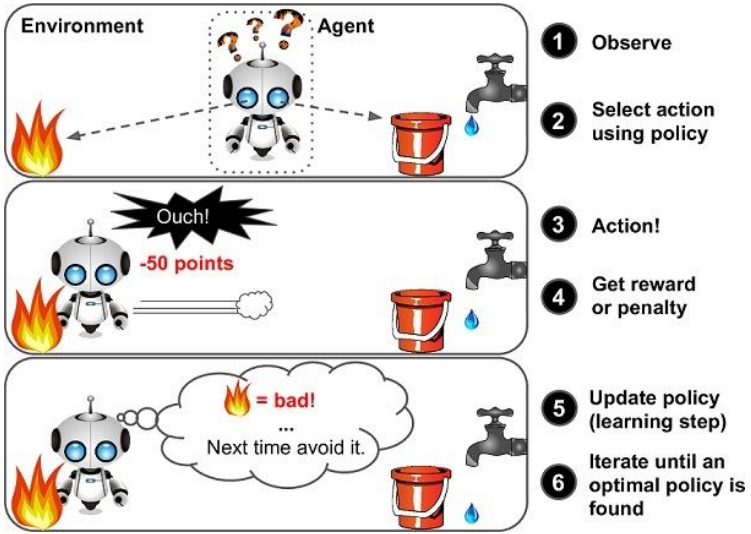
Source: How artificial intelligence will revolutionize the way video games are developed and played

ML : Reinforcement Learning (RL)

RL is the training of machine learning models to make a sequence of decisions.

The agent learns to achieve a goal in an uncertain, potentially complex environment



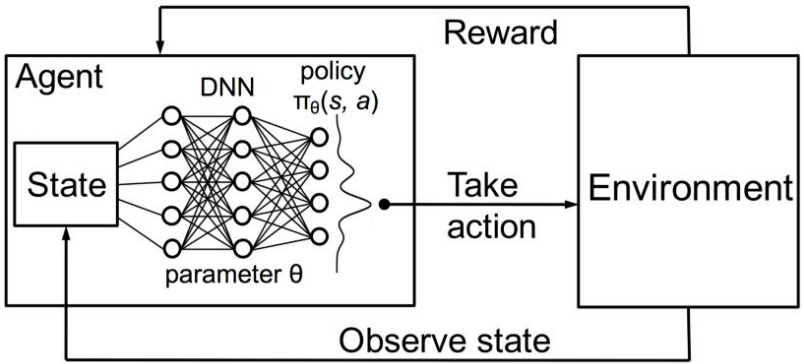


[Flappy Bird reinforcement learning agent in Python]

ML : Deep Reinforcement Learning

DRL combines aspects of reinforcement learning and deep neural networks and is done with two different techniques:

- Deep Q-learning methods aim to predict which rewards will follow certain actions taken in a given state.
- Policy gradient approaches aim to optimize the action space, predicting the actions themselves.



[Spinning Up a Pong AI With Deep Reinforcement Learning]

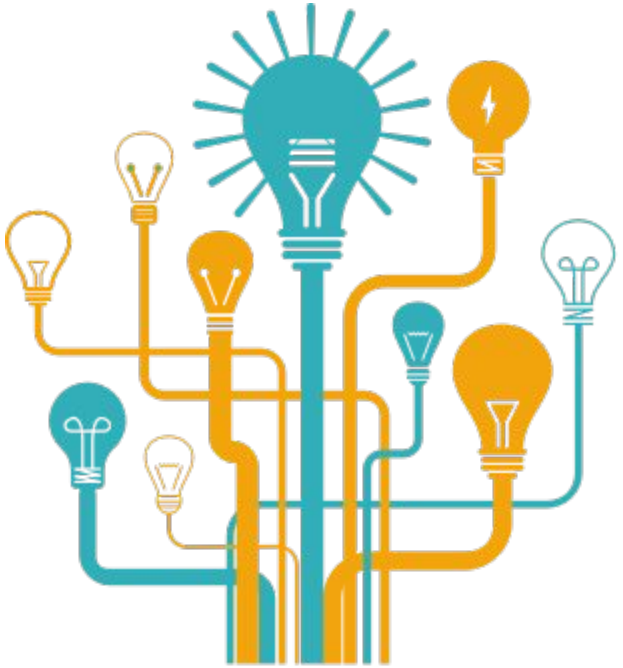
“

I thought AlphaGo was based on probability calculation and that it was merely a machine. But when I saw this move, I changed my mind. Surely, AlphaGo is creative.

Lee Sedol

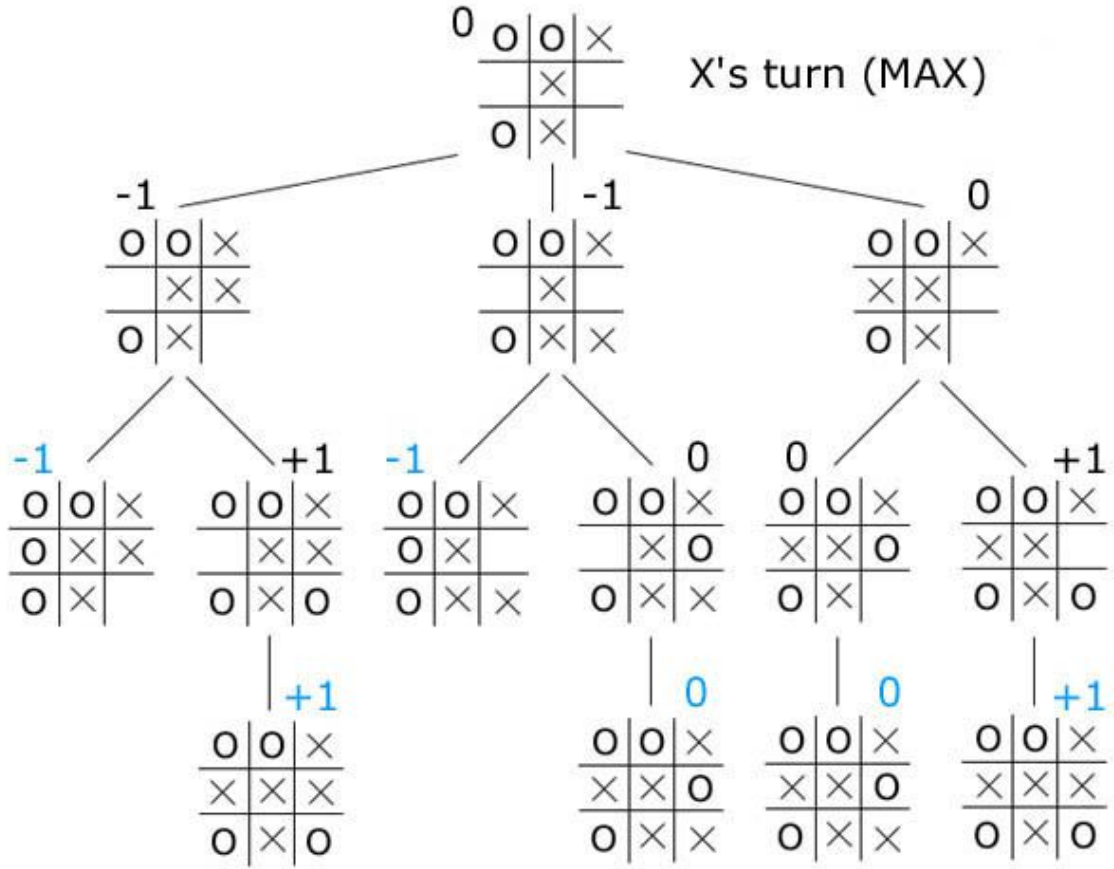
Winner of 18 world Go titles

Thank YOU!



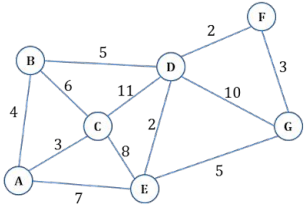
Appendix

Tic-Tac-Toe : Max-Min



[Tic-Tac-Toe in Python]

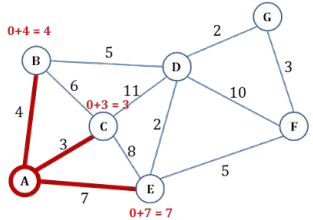
AI pathfinding - Dijkstra's Algorithm



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	∞	
C	∞	
D	∞	
E	∞	
F	∞	
G	∞	

Visited = [], Unvisited = [A,B,C,D,E,F,G]

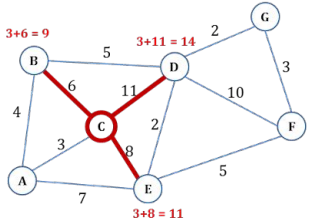
Step 1



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	∞	
E	7	A
F	∞	
G	∞	

Visited = [A], Unvisited = [B,C,D,E,F,G]

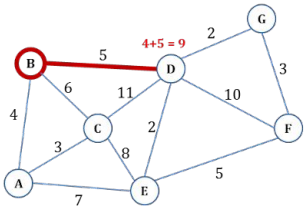
Step 2



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	14	C
E	7	A
F	∞	
G	∞	

Visited = [A, C], Unvisited = [B,D,E,F,G]

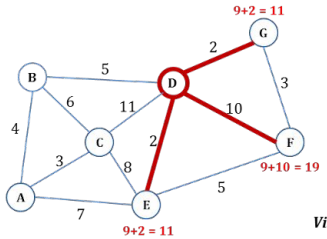
Step 3



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	9	B
E	7	A
F	∞	
G	∞	

Visited = [A, C, B], Unvisited = [D,E,F,G]

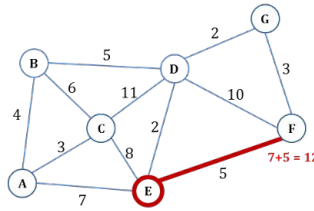
Step 4



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	9	B
E	7	A
F	19	D
G	11	D

Visited = [A, C, B, D], Unvisited = [E,F,G]

Step 5

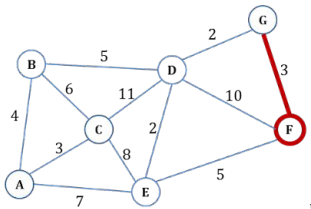


Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	9	B
E	7	A
F	12	E
G	11	D

Visited = [A, C, B, D, E], Unvisited = [F,G]

Step 6

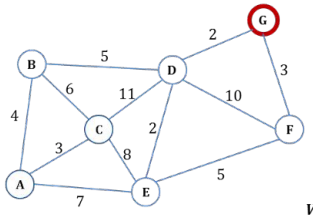
AI pathfinding - Dijkstra's Algorithm



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	9	B
E	7	A
F	12	E
G	11	D

Visited = [A, C, B, D, E, F], Unvisited = [G]

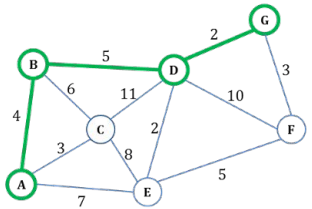
Step 7



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	9	B
E	7	A
F	12	E
G	11	D

Visited = [A, C, B, D, E, F, G], Unvisited = []

Step 8



Vertex	Shortest Distance From Vertex A	Predecessor Vertex
A	0	
B	4	A
C	3	A
D	9	B
E	7	A
F	12	E
G	11	D

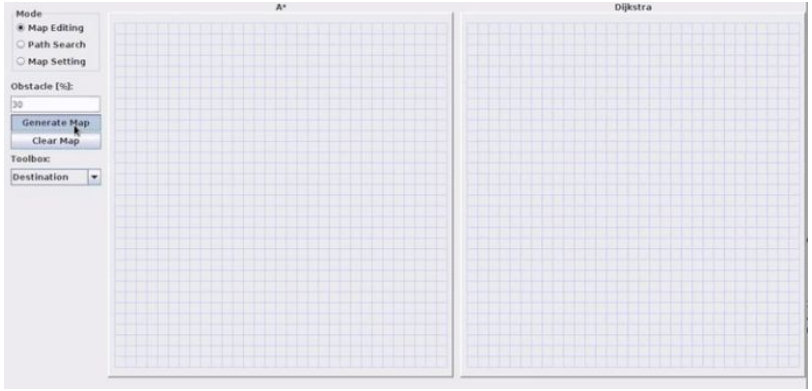
Shortest Path From A To G is {A,B,D,G}

AI pathfinding - A* Algorithm

A* is just like Dijkstra, the only difference is that A* tries to look for a better path by using a heuristic function which gives priority to nodes that are supposed to be better than others while Dijkstra's just explore all possible paths

Compare A* with Dijkstra algorithm

the process takes longer than A*, but it does usually come up with a better path



Implementation of A* in Python