History of AI in Video Games

Augustine Almanza

First Monumental AI in Games

- Chess AI - Deep Blue
    - In 1997 IBM’s Deep BLue defeated chess champion Garry Kasparov.
        - Deep Blue used an algorithm called Minimax.
- Go AI - AlphaGo
    - In 2016, DeepMind’s computer AlphaGo defeated world champion Go player Lee Sedol.
        - Since Go has a higher branching factor than chess, this made it much harder for an AI to solve.
        - AlphaGo used the Monte Carlo Tree Search algorithm.

AI and Games are intertwined in three distinct ways:

- Games have driven the evolution of AI
  - Because they provide precise rules and narrows subdomains that AI conventionally excels at.
- AI is the future of video games
  - Whether it be with robust procedural systems, or narrative directors and games that can create dynamic stories on the fly.
- Games and AI can give us a deeper insight
  - This would be insight into something even more elusive like intelligence itself.

Early Video Game AI

- Early game AI was a blend of static patterns and dynamic breaks from them based on player action.
- The most common of these methods is called a finite state machine.
  - This organizes a game into a series of states each of which has certain instructions.
  - The AI is then driven into one of these states based on the player’s actions.

Examples of Early Video Game AI

- A great example is Pac-Man
  - Each ghost follows a different script in the neutral state.
  - When Pac-Man eats a power pellet, the game changes state.
  - Pac-Man generated deep engaging gameplay that dynamically adjusted to the players input.

VIDEO GAME CARTRIDGE
ATARI® 2600 VCS™
ATARI 2600
PACMAN
ARCADE EDITION
USE WITH JOYSTICK CONTROLLER
© 1983 ATARI, INC. ALL RIGHTS RESERVED. 2646
*PAC-MAN is Licensed by Namco America, Inc

Modern Game AI

- Most modern games use an algorithm called A*
    - Pathfinding algorithm that calculates the best way to travel from A to B.
    - Starts at A and looks at the next open position and selects the closest position to the goal from there.
    - Most modern games combine the use of finite state machines and A* to create compelling gameplay.

Behavioral Tree Scripts

- A variant of the finite state machine
  - Starts with the “root” and branches off to end with “children.”
  - Popularized by Halo 2 to increase complexity of enemy characters.
  - Bungie used this to their advantage to create squad based tactics with enemy AI.

Goal-Oriented Action Programming

- First majorly utilized in F.E.A.R.
  - Through the strips system, this uses planning to let the AI decide how to achieve a given goal.
  - Benefits of this system:
    - Layering of behaviors
    - Decouple goals from strategies
    - Create dynamic novelty

Combinations of AI

- The design of AI in video games is not driven by technology necessarily, but by the demands of crafting new and engaging forms of play.
- Metal Gear Solid uses a new take on finite state machines, combining it with environmental queues.
- Different combinations of AI will be more appropriate depending on what type of game is being made.

Modern Applications of Game AI and Real AI

A Brief Overview of this Section

- Game AI vs Real AI
- Machine Learning Techniques for games
- Applications in Games
- OpenAI
- Applications of AI in Everyday Life

Game AI vs. Real AI

- Game AI
    - “Game AI’ is used to refer to a broad set of algorithms that don’t use “True AI” techniques
    - This is used to generate responsive, adaptive and/or “intelligent” behaviors is non-player characters (NPC’s)
    - Serves to to improve the players experience rather than machine learning or decision making intelligence.
    - Game AI often consists of some rules of thumb (heuristics), that happen to be enough to provide a gameplay enhancing experience.
    - There is a predetermined and limited set of responses to a predetermined and limited set of inputs.

Game AI vs. Real AI

- Real AI
    - “Real” or “true” AI addresses the fields of machine learning which is still used in games, but not as frequently due to the higher level of computing required.
    - A lot of AI in games are very refined versions of game AI, and companies will push their games with
- There still is applications of real AI in games, and we will cover some of the more commonly used.

Machine Learning Techniques for Games

- Deep Learning
  - This branch focuses heavy of artificial neural networks (ANN) that learn to solve tasks.
  - Uses multiple layers of an ANN and other tricks to progressively extract info from inputs.
  - Since this is a complex and layered approach, often requires a powerful machine to train and run on.
- Convolutional Neural Networks
  - These are specialized ANN’s that often analyze image data.
  - Are able to learn patterns which are that don’t depend on location, called translation invariant patterns.

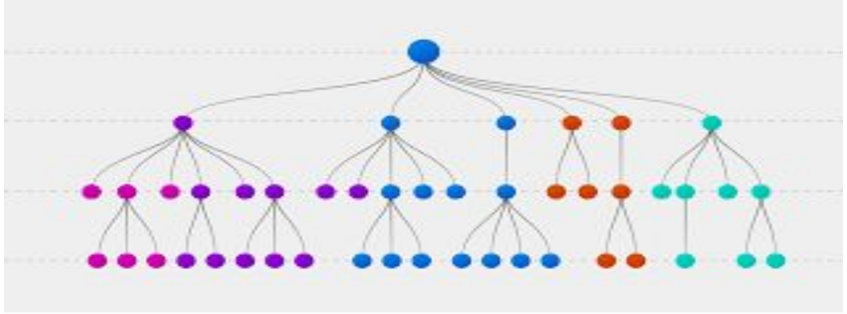
Machine Learning Techniques for Games

- Recurrent Neural Network
  - These are a type of ANN that are designed to process sequences of data in order, one part at a time instead of all at once.
  - They basically run over a given part in the sequence, using current part and memory of previous parts of the same sequence to produce an output.
  - This type of network is very effective with tasks such as speech or text recognition.
- Neuroevolution
  - This is the practice of using both neural networks and evolutionary algorithms.
  - Instead of using gradient descent (very common with neural networks), they use evolutionary algorithms to update neurons in a network.

Machine Learning Techniques for Games

- Reinforcement learning
  - The process of training an agent using rewards and/or punishments.
  - Reward vs punishment will largely depend on the problem
    - Giving positive reward for winning game, or negative one for losing
  - This technique is used heavily with methods such as Q-learning, policy search, and others (outside our scope).
  - This usually has a very strong performance in video games and robotics.

Game AI applications



- NPC’s
  - Scripting (decision tree) is the most common means of control in modern gaming.
  - Often has bugs that need to be worked out such as repetitive and abnormal behaviour, coined “artificial stupidity”
- Pathfinding
  - Very common use, seen most often in real-time strategy (RTS) games.
  - The method for NPC to find a path from one point to another, taking into consideration game elements.

Game AI applications

- Player-experience modeling
  - This can include dynamic game difficulty balancing, which in real time adjusts the difficulty of the game based on players skill.
  - Can also be used to deduce player intent
    - Player pointing weapon/causing something dangerous
    - Gesture recognition
- Procedural content generation
  - Creating elements of game environment like conditions, level, and sometimes music.
- Data mining user behaviour
  - Allows the game designer to get insight on how the game is played.

OpenAI

- “OpenAI’s mission is to ensure that artificial intelligence benefits all of humanity. An important part of this effort is training AI systems to do what humans want.”
- They are a company that designs more than just game AI, but we will specifically talk about OpenAI Five.

OpenAI Five

- A 3 year long project spanning 2016-2019 whose goal was to use the complex MOBA game Dota 2 as a research platform for general-purpose AI.
- The Dota 2 AI learned by playing 10,000 years of games against itself.
- It demonstrated the ability to achieve professional-level performance, boasting an astonishing achievement of a back-to-back win against the world title team, as well as an overall 99.4% win rate (7215 wins, 42 loses)

Dawson Burgess

Applications of AI in Everyday Life

- Self-driving and self-parking cars
- Digital Assistants
  - Google, Siri, Alexa, etc.
- Robots
- Transportation
  - Uber
- Image Recognition software
  - Faces, objects, etc.

Dawson Burgess

Minimax Algorithm

What it is

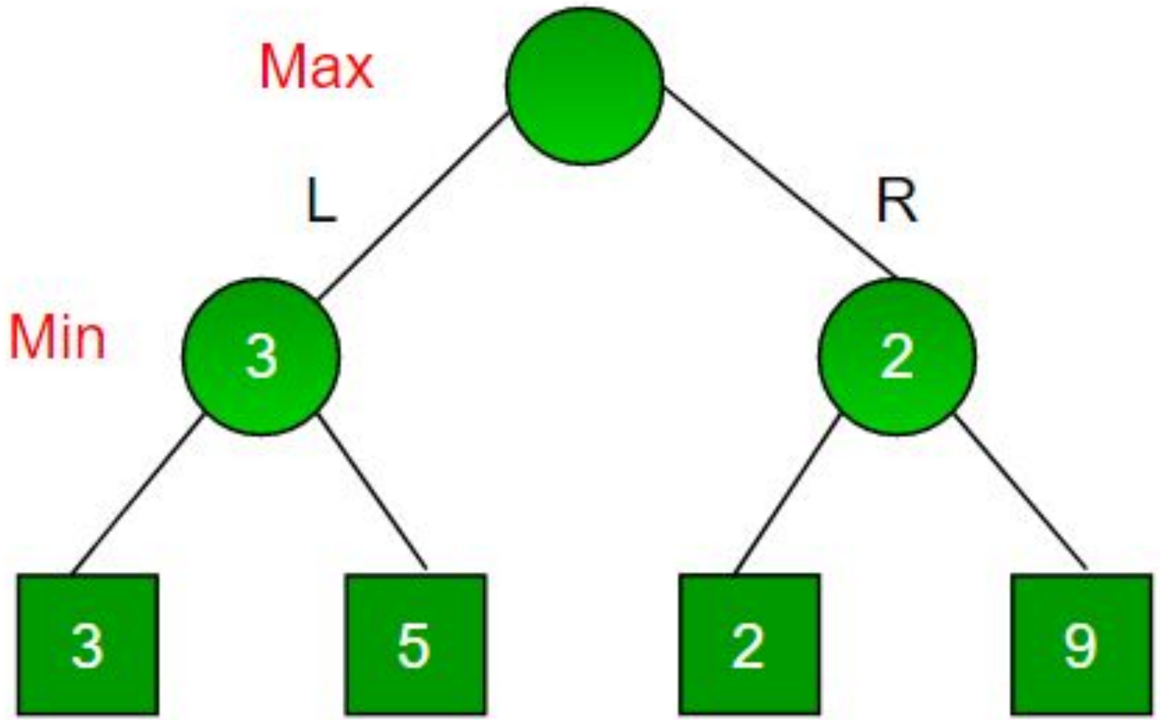
- Algorithm for two person games
- Looks at all possible moves up to a set number of turns ahead
- Looks to maximize their position while the other player looks to minimize it
    - Hence the name minimax
- Assumes both players are playing fully optimally

Where it is typically used

- Best used in games in which all possible moves can be known with 100% certainty
- Best to look between 4 to 5 moves ahead depending on the game
- If you want a pretty slow Chess AI, this is one of the easiest algorithms to implement

What it looks like

- Create a tree of moves
- The maximize player is the root of the tree and each layer represents the next turn of the game
- Each leaf is given a value that represents how good or bad the game state is for the maximizing player
- Then each layer above takes either the max or min value from the layer below (depending on how big the tree is)
- These values are calculated up to the layer just below the root.
- Let’s look at this on the board


Max
Min
L
R
3
2
3
5
2
9

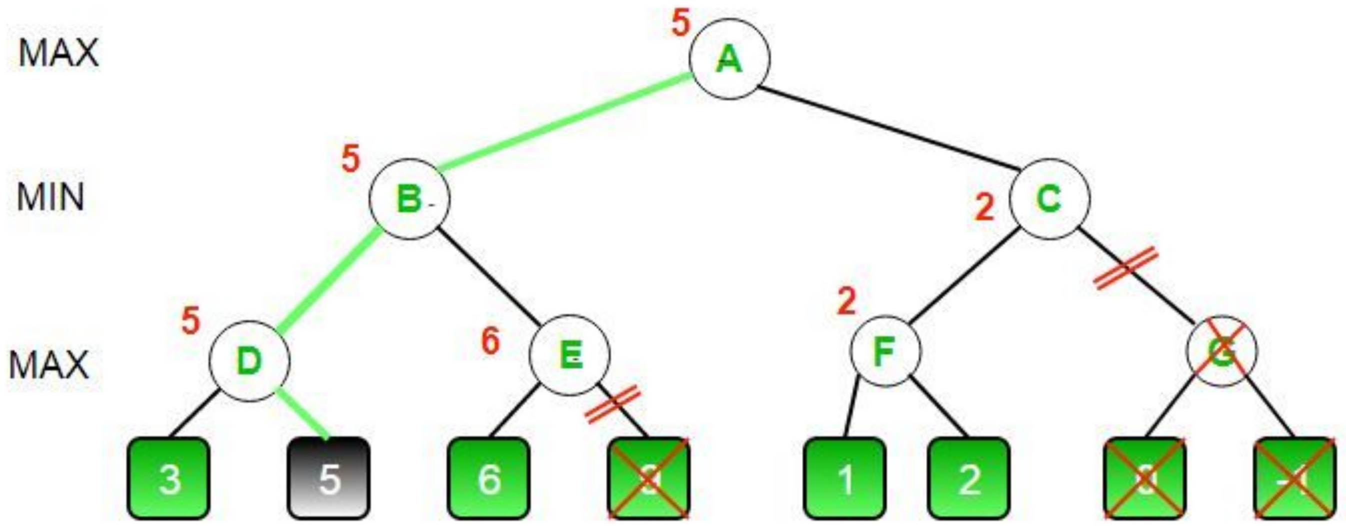
Efficiency

- Since this algorithm relies of DFS, this algorithm is super inefficient the more possible moves and the more moves you want to look ahead
- O(b^m) time, O(bm) space
  - b is the number of possible moves at each turn (the branching factor of the tree), while m is the number moves you look ahead (the depth of the tree)
- Also is affected if other player does not play fully optimally
  - Thus, in order to account for humans being bad at games, minimax must be done more often to account for that.

Kaleb Browning

Let’s make this a bit better

- Use “openings” to avoid unnecessary tree calculations
  - Best used in chess and other longer games
  - Reduces start up time in AI, can even have pre-made trees for after the main opening
- Alpha-beta pruning
  - Way to decrease the amount of branches that we have to look at
  - Best explained by example


MAX
MIN
MAX
5
A
5
B
2
C
5
D
6
E
2
F
G
3
5
6
1
2

Increasing the scope of Minimax + Alpha-Beta Pruning

- As it stands minimax can only be applied to games that have all possible moves known
- Can we apply it to games with unknowns? Maybe.
- Lets ponder an example