AI Concepts to Implementation - CA

**Student full name : Barbara Weltson Leite da Silva**

**Student Number:sba24096**

[**Task 1 1**](#_h57msmso7v1y)

[**Task 2 2**](#_yli7vuh1wq44)

[**Task 3 3**](#_evuu1sjruwro)

[**Reference 4**](#_7bq0cwgo7y89)

[**Github Repo 5**](#_joiheb30s7ql)

# Task 1

**Research two Artificial Intelligence (AI) strategies that have been used to play chess. Describe the theory and concepts used within these strategies and how they relate to AI.**

**The Alpha-Beta algorithm** helps improve decision-making by checking only the most important moves. It introduces two values, called **alpha** and **beta**, which help it figure out the best and worst possible moves without looking at every single option.

* **Alpha** is like the best score the first player (who wants to win) could get. It sets a “minimum” score they want, so if a move is worse than this, the algorithm ignores it since it doesn’t help them.
* **Beta** is like the best score the second player (who tries to block the first player) could get. It sets a “maximum” score they want to achieve, so if a move is better than this, the algorithm ignores it since it won’t help the first player.

When the Alpha-Beta algorithm looks through possible moves, it uses a game tree to explore different paths each player could take. But instead of looking at *every* branch in the tree, it skips over ones that definitely won’t help either player. This is called **pruning**—basically cutting off branches that don’t matter.

As it moves through the game tree, the algorithm keeps updating the alpha and beta values based on what’s found so far. For example, if it finds a move that doesn’t help either player based on alpha or beta, it stops looking at that path and moves on. This way, it doesn’t waste time on moves that won’t change the outcome.

So, by using alpha and beta to focus on the best paths, the Alpha-Beta algorithm can make faster, smarter decisions, ignoring moves that don’t affect the final choice. This makes it a very efficient strategy for finding the next best move in games and AI decision-making.

**Reinforcement learning**: in this model, the AI will learn and improve by playing the game against itself. They use the process of self-play, where the model keeps adding challenges against itself using different versions. In this model, AI will be playing millions of games, and then the model will start to learn slowly which moves and strategies work for winning.

Every result and every move is saved and used to keep training the neural network model, and so each successful move or failure will teach the model on how to make better decisions in future games.

This model is a bit further way from traditional models which autilize pre-programmed rules or analyze past games,

Reinforcement Learning (RL) is a modern and powerful approach in AI. One famous chess engine using this technique is AlphaZero. In reinforcement learning, the AI model learns and improves by playing games against itself, starting with completely random moves. This process is called **self-play**, where the AI continually challenges different versions of itself. By playing millions of games, the model gradually learns which moves and strategies work best.

Every board position and game result is saved and used to train the AI’s **neural network**—the core “brain” of the system. This means that each move’s success or failure teaches the AI to make better decisions in future games. Over time, the neural network becomes very good at estimating which moves lead to winning outcomes, which is essential for complex games like chess. Whenever a new version of the neural network is trained, it’s then evaluated against older versions by playing games against each other.

Each game produces a dataset that includes multiple board positions, and multiple games build up vast datasets that allow the AI to see patterns and refine its strategies. Unlike traditional methods that rely on pre-programmed rules or analyzing past human games, reinforcement learning enables the AI to find creative solutions on its own, adapting to different situations in ways that often go beyond traditional chess engines.

Using this reinforcement learning algorithm, AlphaZero emerged as a powerful system. In fact, AlphaZero defeated Stockfish, the reigning world champion chess engine, which is free and open-source. Out of 1000 games, AlphaZero won 155 games and only lost 6, showing the strength of this learning approach.

This algorithm allows AI to learn and improve on its own, which is an amazing technique for an AI model

# Task 2

**(b)AI has been essential in computer games since the 1960s. Provide an example of two AI strategies that are used in modern games. You should identify the algorithm being used, provide a high-level description of the algorithm and provide an example of a game where this algorithm has been implemented.**

1 - One of the AI strategies used in video games today is the Finite State Machine (FSM) in game development, especially for Non-Playable Characters (NPCs). An FSM uses algorithms that respond to specific situations or "states" in the game, using things like switch statements and Enums to control how characters behave. These algorithms help NPCs perform actions based on their current state, like patrolling, attacking, or being idle. Recently, new AI technology has been added to games, which allows NPCs to have more dynamic and realistic conversations with players. Unlike the traditional FSM, which only uses a set list of actions, this new AI can generate responses on the fly, giving players more options and making interactions feel more natural. Games like *Cygnus Enterprises* and *Niantic* use this type of technology to create more realistic NPCs and provide players with a wider range of choices in how they interact with the game world. As AI continues to improve, NPCs will likely become even more responsive, creating a more engaging and personalized experience for players.

2 - The second example I’ll be adding here is quite recent, related to an AI model that is simulating Counter Strike inside a neural network . On this strategy token prediction is used for Transformer models, and is applied by the DIAMOND model to simulate *Counter-Strike: Global Offensive* (CS). On this, player actions and environmental factors are treated as tokens, and then the model is used to predict future behavior based on prior actions. The model learns complex game dynamics such as player movements, weapon usage, and tactics, allowing it to simulate real-time decision-making in Counter Strike, which is a very famous FPS game.

# Task 3

**SONY has developed a robot puppy named AIBO. You can find information about this robot at https://us.aibo.com/.**

* **a) Discuss the AI characteristics that you think the AIBO exhibits. [15 marks]**

AIBO clearly has a lot of similarities with a real dog, including some impressive dog sounds and actions that dogs would do in real life. AIBO is able to recognize its owner's voice and perform little plays based on the owner’s requests and sentiments, giving the owner the slight feeling of having a real dog. It can also understand basic commands, so it understands natural language and responds to verbal commands from its owner.

AIBO is also an independent learner and will grow and understand the environment based on everyday routines and relationships, so there is an autonomous behavior that it will acquire with time as it learns more and more about its own preferences, such as how it prefers to be petted. This autonomy allows AIBO to become very close to its owner's habits, and AIBO can become “attached” to its owner due to this.

AIBO also comes with several technological features, such as capturing moments and uploading them to a database or even taking pictures when requested. This sensory awareness lets it monitor and interact with its environment, and the cameras allow it to detect familiar faces, recognize its owner's voice, and respond uniquely to different people.

Another notable feature is AIBO’s simulation of emotional responses. AIBO can display signs of excitement and affection depending on the situation. These emotions are designed to make AIBO appear more like a real life dog so the owner feels a greater connection to it.

AIBO uses pattern recognition and machine learning to adapt its responses based on repeated interactions.

**b) AI Characteristics that AIBO Still Lacks**

I believe the main characteristic AIBO will lack is that “soft skills” or “emotion.” While it can mimic a real dog, it doesn’t convey to the user the genuine feeling of companionship that a real dog would provide. AIBO follows a set of rules for how to act in various situations, but in unique scenarios where a real dog might react unexpectedly, AIBO’s responses are more predictable, reminding its owner that it’s ultimately not a real pet.

Another big issue with AIBO is that it often seems repetitive and isn’t truly able to learn. It lacks deep learning capabilities that would allow it to master new tricks or behaviors on its own, which can make interactions feel less dynamic. Instead, AIBO relies on a limited set of pre-programmed actions, so it doesn’t evolve in the same way a real pet would by adapting to its owner’s personality and habits.

AIBO also lacks a nuanced understanding of context and emotion. While it recognizes voices and can respond to commands, it cannot interpret the emotional meaning behind those commands. For example, if its owner is sad or excited, AIBO doesn’t adjust its behavior to reflect empathy or understanding, which diminishes its role as a true companion.

Finally, AIBO doesn’t have the ability to self-improve or refine its own actions over time. While it remembers frequent commands or interactions, it doesn’t naturally evolve or learn from its experiences in a meaningful way. This limitation makes it challenging for AIBO to develop a deeper, more personalized relationship with its owner, as it doesn’t adapt in the same way a real pet would.

# Reference

aibo. (n.d.). *aibo*. [online] Available at: https://us.aibo.com/.

Bex Tuychiev (2024). *Implementing the Dijkstra Algorithm in Python: A Step-by-Step Tutorial*. [online] Datacamp.com. Available at: https://www.datacamp.com/tutorial/dijkstra-algorithm-in-python.

Chessprogramming.org. (2010). *Alpha-Beta - Chessprogramming wiki*. [online] Available at: https://www.chessprogramming.org/Alpha-Beta.

inworld.ai. (2024). *AI NPCs and the future of gaming*. [online] Available at: https://inworld.ai/blog/ai-npcs-and-the-future-of-video-games.

Jagdale, D. (2021). Finite State Machine in Game Development. *International Journal of Advanced Research in Science, Communication and Technology*, 10(1), pp.384–390. doi:https://doi.org/10.48175/ijarsct-2062.

Klein, D. (2021). *Neural Networks For Chess*. Independently Published.

Silver, D., Hubert, T., Schrittwieser, J. and Hassabis, D. (2018). *AlphaZero: Shedding new light on chess, shogi, and Go*. [online] Google DeepMind. Available at: https://deepmind.google/discover/blog/alphazero-shedding-new-light-on-chess-shogi-and-go/.

# Github Repo

https://github.com/babiweltson/ca-ai-concepts-to-implementation