CHESS ENGINES ON GPU'S

MSc Advanced Computer Science

Project Specification Report

Supervised by Dr.Ben Mora

CSCM10

2021-2022

2036334

Literature Review

Chess as a game is played with strong strategic moves. The advancement in technology has improved to a major extent to which a Computer is able to define the moves made by the opponent and is able to get a calculative response so fast and accurate that is unimaginable. The approaches in which enable GPU's to play chess and the algorithms which are designed through every step of the way is a very technical way to observe and grasp how exactly the engines gain from the algorithm and process the futuristic outcome. The understanding and the way in which the chess engine understands and categorises is just impeccable. This report defines the Aims and Objectives of my project, the motivation I had to take up this particular project, the description of my project, the different components and deliverables in my project, the research methodology I used, the project plan, risk analysis and finally the Timeline.

Chess is a board game played between two players. The current form of the game has been developed from the Indian origin board game known as Chaturanga. The game is played on a chess board with 64 squares which is in the format of a 8*8 grid. Each player has 16 elements on the board which are as follows, 1 King, 1 Queen, 2 Rooks, 2 Bishops, 2 Knights and 8 Pawns. The main objective of the game is to strategically play it in a way in which the opponent has no moves left to defend the King, this when done is called as checkmate. However, there are numerous ways in which the game may end in a draw. Chess has many connections with the field of Computer Science, Mathematics and Psychology. Scientists began to create modern computer which would learn the way of the game and play with a human.

GPU's were initially designed to digitally process a image through specific algorithms. Even though it is proven that GPU's have more raw computing power than a CPU, GPU's need a specialized and parallelized way of programming. However a set of algorithms which use the concept of machine learning which is deep learning was the only methodology which worked very well with the GPU architecture. There are basically three approaches on how to use a GPU for chess which are as follows: Firstly, it can be used as an accelerator in Leela Chess Zero(LC0) which uses a concept of deep neural network which not similar to the usual chess programs which use the concept of linear function approximation. Secondly, Offload the search in Zeta. Zeta is a opensource chess engine which is written in C and uses ports which link up to the alpha beta search algorithm. Zeta uses ports and has 64 GPU threads which are coupled to one which helps in move generation, move picking and evaluation. Lastly, as an hybrid in perft_gpu, which is basically a brute force program which counts the leaf nodes and explores a tree (Ankan,2017).

There are different types of chess algorithms which help the chess engines to work faster and efficiently. The Minimax algorithm is a recursive algorithm in which we assume that both the white and black players have the best moves, however the greater priority is given to maximizing white's score. This algorithm gives one player a certain chance of winning. The depth-first search takes place after which the leaf nodes are evaluated. All the moves which are calculated by the maximizing player are denoted as positive affinity and all the moves made by the minimizing player are denoted as negative affinity. The minimax function return a heuristic value of the leaf nodes and favours the score of the maximizing player. Based on this observation, max(a,b) will be equal to -min(-a,-b). The code treats both maximizing and minimizing player separately. Minimax can be simplified into a negamax algorithm. This actually leads to combinatorial game theory which was developed by John Horton Conway.

Negamax Algorithms use similar trees such as game trees as used in Minimax algorithms. But here the root node will take the score from its next immediate child node. The child node which depicts the high score from the root node gives the best possible move to play. The maximum value is always

searched by the negamax algorithm and that is why for player B it would always be the minimax score is always the negation of the negamax score obtained.

Alpha-beta pruning always decreases the number of nodes which the negamax algorithm evaluates in its specific search tree. Basically Alpha-Beta are the upper and lower bounds for the child node. When the pruning portion takes place, the values which are falling outside the alpha-beta bounds are eliminated, whereas the values which fall in between are given the true value. These values do not affect the root node at all. The complication here starts when the transposition table is added because it becomes difficult to track a node's value because the value may not be the node's true value.

It is just amazing to see how a machine is able to learn, understand and outsmart the opponent by the help of the algorithms laid out for it. This speaks for the futuristic developments which can be made in this particular field and also in the field of Artificial Intelligence.

Project Aims and Objectives

The Aim of my project is to explore the different chess engines and how the algorithms enhance the working of it to a greater extent and also how GPU's can be utilised to work through in chess engines. I have also explored the recent machine learning breakthroughs in the field of chess. The objectives of my project were to

1. Explore Chess as a game and the rules of it.
2. Enhance knowledge of the different types of chess engines like Stockfish.
3. Find out more about the Chess Algorithms such as Negamax, Minimax and Alpha beta pruning.
4. Exploring Alpha go and Alpha Zero.
5. Exploring the recent Machine learning breakthroughs in the field of chess.
6. Analysing the suitability of chess engines on GPU's.

Motivation

First and foremost Chess is one of the most intellectual games that anyone can play. It involves using mental power, concentration and focus. Most of the professional chess players think further ahead even before they make a single move. The concept of chess and how it is played requires a good amount of intelligence to play it well, the Chess engines which are built and designed surprise us by letting us know the power of Artificial intelligence and Deep neural networks.

Nothing can beat the art of how a machine learns all by itself and also is able to outsmart the human opponent playing with it. In addition to this, GPU's were primarily intended for faster image processing but it is amazing to see even a GPU can also be used to play chess. Leela Chess Zero has proven that the Monte Carlo Tree Search combined with Deep learning methodology will blend in with the GPU architecture.

The fact that combing all the chess algorithms alongside GPU to create the chess engines is very intriguing and in fact speaks for the future on how strong the machines are capable of.

Project-Description

My project revolves around exploring the different types of Chess engines, Chess algorithms used and also the recent Machine Learning breakthroughs in the field of chess which is very interesting to learn about.

Since there are many Chess engines, the classification of these engines take place on two factors which are Speed in which it calculates the opponents moves and the iteration of the tree and the other factor is Accuracy in which it evaluates the chess board. Stockfish is a Chess engine which uses bitboards to form similarity to a chess board and further uses logical operations based on a single move made by the opponent. The tree is built after it gets the evaluation of all the possible moves at that particular time. This is an example of a stockfish code which is to get a square attacked by a bitboard of pawns.

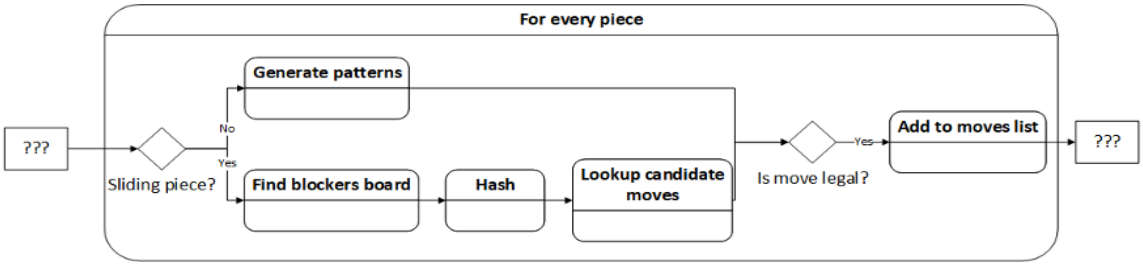
Template<Color C>

Constexpr Bitboard pawn_attacks bb(bitboard b) {

Return C == White ? shift<north_west>(b) | shift<north_east>(b)

: shift<north_west>(b) | shift<north_east>(b);

} (Antoine Champion,2021)



(Fig1, Representation of evaluation done by stockfish for every piece)

Here, Stockfish is creating a blockers board which is the result of the logical operation of Complete board at that moment of time AND the attack mask. Stockfish creates blockers board to hold the index of array of the opponents precalculated moves. Hash key is used to reduce the amount of memory stored in the original bitboards. The sliding pieces are the ones which have an indefinite movement in a particular direction till the edge of the board unless obstructed by any other piece.

AlphaZero however is a highly advanced chess engine which was developed by Deepmind, acquired by Google. This engine was just taught the rules of the game and it ended up playing 44 million times by itself. It has the power to calculate upto 70 million moves per second. This engine is not open to the public. It runs on custom built hardware usually known as Google Supercomputer and can easily beat Stockfish.

On the other hand, AlphaGo works on MCTS(Monte Carlo Tree Search) Algorithm. It builds upon four different steps which are Selection, Expansion, Evaluation and Backup.

The Minimax algorithm is a recursive algorithm in which we assume that both the white and black players have the best moves, however the greater priority is given to maximizing white's score. This algorithm gives one player a certain chance of winning. The depth-first search takes place after which the leaf nodes are evaluated. All the moves which are calculated by the maximizing player are denoted as positive affinity and all the moves made by the minimizing player are denoted as negative affinity.

Negamax Algorithms use similar trees such as game trees as used in Minimax algorithms. But here the root node will take the score from its next immediate child node. The child node which depicts the high score from the root node gives the best possible move to play.

Alpha-beta pruning always decreases the number of nodes which the negamax algorithm evaluates in its specific search tree. Basically Alpha-Beta are the upper and lower bounds for the child node. When the pruning portion takes place, the values which are falling outside the alpha-beta bounds are eliminated.

Matthew Lai from Imperial College of London has recently created a Artificial Intelligence based system called as Giraffe. It has taught itself to play chess much more like a human being by evaluating positions and in a very different way as compared to the conventional chess engines. The concept behind this is of a neural network, tends to work as close to similar as the human brain. It consists of layers of nodes which are connected to each other in a way in which it automatically trains itself and self-tunes the best possible move. It is equivalent to that of a FIDE International Masters status which is in turn equivalent to 2.2 of the top chess players in the world.

Maia was developed from the roots of the code of Leela Chess Zero which favours accurate human calculations. Maia works very differently as compared to the usual chess engines, here it favours the highest possibility of the human move. The data to teach Maia was fetched from LiChess which is an online chess program. Now Maia is tuned to different levels of gameplay.

Deliverables and Components

My main Deliverables and Components were the different types of chess engines such

- Stockfish
- Alpha Go
- Alpha Zero.

The Chess algorithms used to enhance the engines such as

- Alpha-beta pruning
- Negamax algorithm
- Minimax algorithm were one of my important components as well.

➢ Studying and exploring the recent machine learning breakthroughs was one of the huge part in my project.

Research Methodology

The type of research methodology I have used is Exploratory research which involves doing a study which explores the deeper roots of that particular topic. All the information in my project has been obtained by me after doing a enhanced and in depth study of all the topics involved. Breaking down each topic and splitting it into different parts and then studying about it has given me the best information I could possibly achieve for my project.

Project Plan

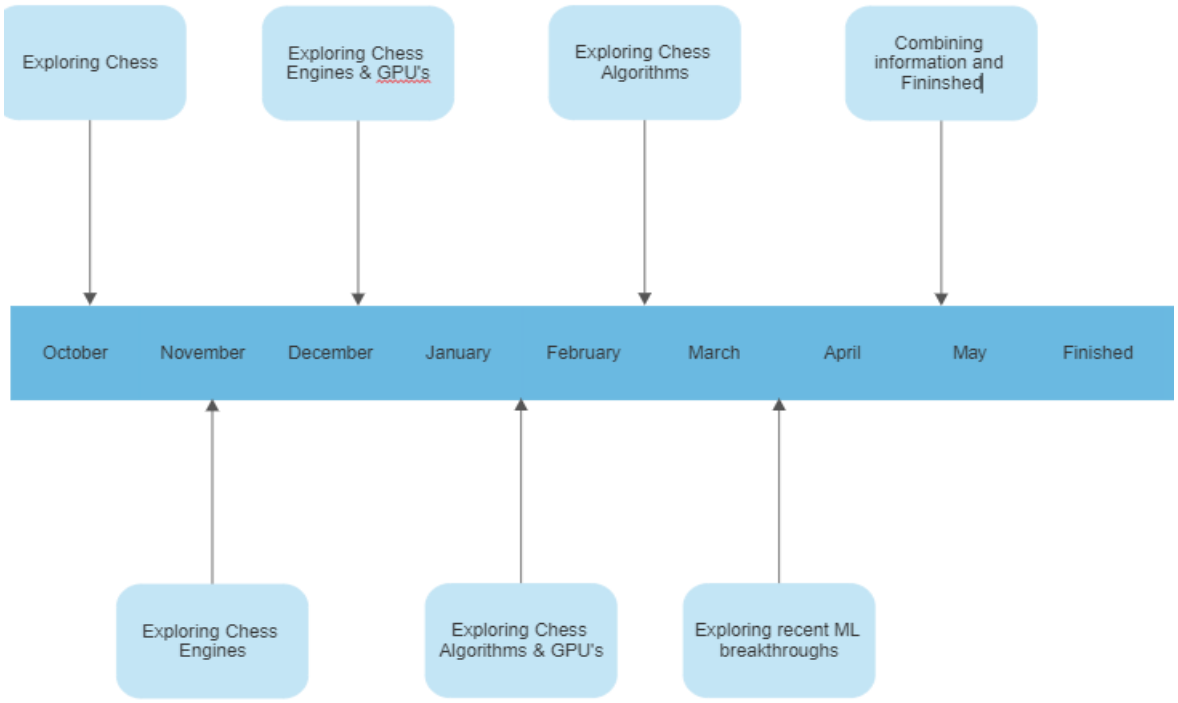
The first step of my project plan was to break down all the components I had so that a in depth study could be performed on each of the component so that more information could be gained. Since my project revolved around the concept of chess, I started by finding out about the game itself and how it was evolved. After this I focused on GPU's and the approaches in which GPU's could be used for chess. After finding out about the approaches, I started exploring the different types of algorithms which is huge part of my project and for the chess engines as well. After this I combined all the

information and connected all the pieces of information. Finally, I started exploring more about the recent machine learning breakthroughs in the field of chess.

Risk Analysis

Even though the chess engines are designed very intricately, there are still room for errors and blank spots such as the first line of the code written might not always be the best first move to make. Furthermore, when a human plays with a chess engine there are numerous factors which are involved such as mental pressure whereas machines do not come across these factors. For chess engines it is basically king vs king game, and also throws a probable move which would outsmart the opponent.

Timeline with task descriptions and mile stones



References

AlphaZero - Chess Engines. (2020). Chess.Com. Retrieved 21 February 2022, from https://www.chess.com/terms/alphazero-chess-engine

arXiv, E. T. F. T. (2020, April 2). *Deep Learning Machine Teaches Itself Chess in 72 Hours, Plays at International Master Level*. MIT Technology Review. Retrieved 23 February 2022, from

https://www.technologyreview.com/2015/09/14/247956/deep-learning-machine-teaches-itself-chess-in-72-hours-plays-at-international-master/
Champion, A. (2022, January 1). *Stockfish: In-Depth look at a chess engine (1) | Towards Data Science*. Medium. Retrieved 20 February 2022, from https://towardsdatascience.com/dissecting-stockfish-part-1-in-depth-look-at-a-chess-engine-7fddd1d83579#:%7E:text=Stockfish%20is%20actually%20performing%20the,blocking%20pieces%20or%20discovered%20checks
Deep Learning - Chessprogramming wiki. (2021). Chessprogramming. Retrieved 25 February 2022, from https://www.chessprogramming.org/Deep_Learning
Djurhuus, R. (2014). Chess Algorithms Theory and Practice. *Chess Algorithms Theory and Practice*. https://www.uio.no/studier/emner/matnat/ifi/INF4130/h14/undervisningsmateriale/chess-algorithms---theory-and-practice_ver2014.pdf
GPU - Chessprogramming wiki. (2022). GPU. Retrieved 23 February 2022, from https://www.chessprogramming.org/GPU
Hui, J. (2018, May 17). *AlphaGo: How it works technically? - Jonathan Hui*. Medium. https://jonathan-hui.medium.com/alphago-how-it-works-technically-26ddcc085319
Knight, W. (2021, February 13). *A New Artificial Intelligence Makes Mistakes—on Purpose*. Wired. Retrieved 25 February 2022, from https://www.wired.com/story/new-artificial-intelligence-mistakes-purpose-chess/
Leela Chess Zero - Chessprogramming wiki. (2021). Leela_Chess_Zero. Retrieved 25 February 2022, from https://www.chessprogramming.org/Leela_Chess_Zero
perft(15) - Page 1 - TalkChess.com. (2017). Chesstalk. Retrieved 26 February 2022, from https://www.talkchess.com/forum3/viewtopic.php?t=64983&start=4#p729152
Swansea University. (1920). [Photograph]. https://www.swansea.ac.uk/
Wikipedia contributors. (2022a, January 22). *Negamax*. Wikipedia. Retrieved 23 February 2022, from https://en.wikipedia.org/wiki/Negamax#:%7E:text=Negamax%20with%20alpha%20beta%20pruning,An%20animated%20pedagogical&text=Alpha%2Dbeta%20pruning%20can%20decrease,use%20with%20the%20minimax%20algorithm.&text=Alpha%20(%CE%B1)%20and%20beta%20(,at%20a%20given%20tree%20depth
Wikipedia contributors. (2022b, February 27). *Minimax*. Wikipedia. Retrieved 24 February 2022, from https://en.wikipedia.org/wiki/Minimax#:%7E:text=The%20Minimax%20algorithm%20helps%20find,B's%20own%20chances%20of%20winning)
Pogonina, N. (2012, January 25). *Chess Engines' Evaluations*. Chess.Com. Retrieved 20 April 2022, from https://www.chess.com/article/view/chess-engines-evaluations