Parallel Chess AI
Proposal

Ishaq Haj Hasan
Andrew ID: ihajhasa
B.Sc. in Computer Science

Sameer Ahmad
Andrew ID: sjahmad
B.Sc. in Computer Science

URL Github: /ihajhasa/Parallel_Chess_AI

Summary: We will be implementing a Chess AI using Alpha-Beta Pruning Trees to run on NVIDIA GPUs GHC clusters. Our Parallel Chess AI will be able to instantiate

Background: The programming complexity that software developers face when creating models/programs that simulate chess are quite extensive. This is due to the fundamental attributes that a game of chess comprises of. Complementary to popular opinion, Chess is a game of "perfect information" where each state of the game leading upto and including the current state are known by both players involved in the game. The realization that this invokes a whole new layer of computational complexity to accurately model and predict that subsequent moves of our opponents is exactly why it is essential to learn parallelizing techniques for this game.

The Challenge: There is a specific reason as to how and why parallelizing sequential chess algorithms poses a challenging task. First and foremost, most seqeuential chess implementations are based on recursive alpha-beta pruning techniques. As we have seen throughout assingments 3 and 4, recursive tree generating algorithms are indeed quite difficult to parallelize especially since they are built from scratch (solely from initial data parameters). But a positive aspect to this is that a majority of the computational complexity that arises from a duration of a chess game involves optimal tree searching tasks that can be exploited under parallel environments. Therefore, our approach to achieving remarkable speed ups is essentially through this very reason.

Resources: [**Techniques to Parallelize Chess**]
Guidry, M., McClendon, C. (n.d.). Techniques to Parallelize Chess. Retrieved October 30, 2019, from https://pdfs.semanticscholar.org/8e1c/9f70aa4849475199e26ccd3e21c662e2d801.pdf.

[**Parallel Alpha-Beta Pruning of Game Decision Trees**]
Steele, K. (1999). Parallel Alpha-Beta Pruning of Game Decision Trees: A Chess Implementation. Retrieved October 30, 2019, from https://students.cs.byu.edu/snell/Classes/CS584/projectsF99/steele/report.html.

Goals and Deliverables:

1. **PLAN TO ACHIEVE:**
    - A Sequential and Parallel Chess AI over Alpha-Beta Pruning.
2. **HOPE TO ACHIEVE:**
    - Incorporating a machine learning aspect to the program to learn the more games it plays

Platform Choice: A nice property of utilizing CUDA enabled machines is the simplicity of modularizing parallel programs as tools to be used by program.

Schedule

Week	**Task 0**	**Task 1**	**Due**
1	Read about current sequential Chess AI solutions	Begin developing a sequential implementation	Sequential Implementation of Chess AI
2	Read about strategies and chess AI techniques	Begin working on parallel implementation	The most basic functional parallel implementation
3	Test out multiple parallelizing strategies and try to optimize parallel implementation		
4	Further work on parallel implementation (final touches)	Begin writing up analysis on program performance	Finished Parallel implementation and Project Report and started working on poster.
5	Finalize poster		