Srishti Srivastava (srishtis)

Summer Kitahara (skitahar)

15-418

Scrabble Bot

Project Proposal

Website

http://scrabble-bot.weebly.com

Summary

We are going to create a bot for a 3D variation of Scrabble that computes the best move possible in parallel. For our implementation, we will use OpenMP and run on the Xeon Phi's on the latedays cluster.

Background

Scrabble is a popular board game where each player takes turns placing tiles with a letter and variable number of points on a board to form a word. We aim to create a bot to play a three-dimensional extension of Scrabble, referred to as *Upwords*, where players can also stack letter tiles on top of each other to replace letters in words on the board. The compute-intensive aspect for the bot is finding the best set of letter tiles to place on the board. Therefore, we will focus on parallelizing this aspect. We already have a serial implementation of "Scrabble With Stuff" from 15-214 to work from, which will make our project more manageable for the six week time frame.

The Challenge

The basic Scrabble problem is challenging because there are many rules and variables for an AI to manage when coming up with the best move. For instance, the word and its placement must be valid, it must overlap with an existing word, it must maximize the point values of the letters and it must take advantage of any possible power-ups, like double word score, on the board. The 3D Upwords extension also has the added wrinkle of stacks of letters in certain places on the board to overwrite letters to create another valid word. These stacks have point values that must be stored intelligently to avoid computing their scores over and over.

From a parallelization point of view, we anticipate that it will be tricky to manage memory and cache misses because optimal words can be found anywhere on the board, so there is limited data locality. Also, the number of potential moves increase exponentially as the number of words played on the board increase. Finally, the playable letters held by the AI keep changing so the benefits to storing past computations of best moves are somewhat limited.

We expect there to be an interesting trade-off between communication and computation. On one hand, if multiple threads are computing optimal moves for a given board state, it might be valuable to store portions of these computations in global memory for other threads to use to avoid inefficient recomputation. On the other, communication overhead might be too high and it might be more advantageous for each thread to just recompute the scores of previously-seen moves. Additionally, managing workloads across threads may be an interesting problem because some potential words will interact much more heavily with the existing board state than others

Goals and Deliverables

Planned Goals

Because we have starter code, including a sequential algorithm and basic GUI from the 15-214 “Scrabble With Stuff” assignment, we believe we can reach our milestones. We hope to first implement a sequential 2D Scrabble by translating our starter Java code to C/C++. Then we will parallelize our Scrabble implementation by computing all possible moves the AI can take in parallel. We hope to attain 30x speedup, similar to the speedup achieved in Assignment 3, which has a similar problem space using OpenMP on the latedays clusters. While continuing to optimize our parallel solution, we will also take the basic Java GUI from the “Scrabble With Stuff” assignment and modify it as we see fit. Then, we will link it to our C/C++ code. After this, we will extend our parallel implementation to work on a 3D version of Scrabble, *Upwords*, and also add this to our GUI. Lastly, we will make our poster to show off our Scrabble bot.

Stretch Goals

Beyond 3D *Upwords* implementation, we may add additional features, rules, and power-ups. Or we could extend our solution to other grid-based games.

Demo

On December 12, we will show our project by having an interactive demo where people can challenge our bot in Scrabble.

Platform Choice

We choose to use the latedays cluster. We will use OpenMP to write the parallel implementation and Java for the graphical user interface. We think the board size and data will be relatively small so it will be safe to store in local memory. We are not using message passing, because it would be expensive to pass the board around, and only sending the best score wouldn't be worth the overhead. Similarly, we chose to not use CUDA because dealing with passing board information between global and device memory will also generate additional overhead costs and the board is small anyway.

Schedule

Week 1 (10/30 - 11/5)

Submit project proposal on 11/1.

Week 2 (11/6 - 11/12)

Have sequential Scrabble algorithm working.

Week 3 (11/13 - 11/19)

Take a first pass at parallelizing Scrabble.

Week 4 (11/20 - 11/26)

Touch up GUI from 15-214. Link GUI to output from our C/C++ code. Continue to optimize our parallel algorithm. We hope to attain 30x speedup, similar to the speedup achieved in Assignment 3, which has a similar problem space using OpenMP on the latedays clusters.

Week 5 (11/27 - 12/3)

Finish 3D *Upwords*. Stretch goal to add additional rules and features to our 3D Scrabble variation.

Week 6 (12/4 - 12/12)

Spruce up GUI and present poster.