Project Proposal
15418 Parallel Computation

Daniel Zeng, Brad Zhang

November 9, 2022

TITLE: Perfect Clear Solver, Daniel Zeng, Brad Zhang

URL: `https://github.com/dlzeng243/15418f22_Final_Project`

SUMMARY: In Tetris, a Perfect clear is when we clear the matrix and leave no blocks remaining. We are going to create a Perfect Clear solver given any 10 pieces and attempt to parallelize it with OpenMP.

BACKGROUND:

We will start off with implementing a Perfect Clear solver which takes in 10 Tetris pieces and attempts to tile a 4x10 grid with them. An example is shon above given the following sequence: [I, I, J, L, L, O, S, T, T, Z]. This won't necessarily conform to the actual Tetris rules of clearing lines and rotation, but it will be a preliminary start to adding these rules to it.

As we can see, there are many positions that a single piece can take, and this would impact on the number of possible placements for the next piece. Thus, this gives us an opportunity to parallelize by splitting them into different cases that we want to look at.

CHALLENGE:
Our first implementation is easily solvable with a search tree. However, the number of placement positions creates a large branching factor, so our first objective/challenge is to find ways to prune the branches.

It's not entirely clear how to do so; furthermore, there may be opportunities to find speedup by communicating information to different threads at certain times.
The workload is generating and checking all the nodes of a tree. As a result, each task is dependent on the task regarding the previous piece placed. As we are planning on using OpenMP, the only memory shared is the current state of the matrix and the remaining pieces to be used. Locality largely depends on the relatedness of the tasks a thread executes.

RESOURCES:
We don't have any base code to start from. There are existing perfect clear finders but we won't be using any directly (they can be found here and here). We might end up using the PSC machines to see if our approach is scalable and the speedup is significant, but we aren't planning on using any other machines to test our parallelization code.

GOALS:
Need To Achieve:

- A working, correct PC solver, single-threaded. This is entirely achievable and must happen since its logic is core part of our project.
- A PC solver with at least 2x speedup on GHC (8 cores). This should be achievable since we have 8 threads so 2x speedup shouldn't be that hard.

Plan To Achieve:

- Add Tetris rotation rules to our single-threaded and parallel implementations. This is achievable if we are given time to implement it.
- A PC solver with at least 4x speedup on GHC. This should be achievable since we have 8 threads so 4x speedup shouldn't be that hard, but we're unsure of how much speedup we can get.

Hope To Achieve:

- Enforce piece ordering (first piece must be placed on the bottom, etc.). This is achievable given time to implement it, but it will greatly increase the complexity of the problem and simulation..
- Add incomplete information (only know the next 5 pieces) and create parallel algorithm to maximize probability of creating a PC. This adds a lot of complexity to our original project given we could just generate a tree and attempt to parallelize over the branches.

Demo: We plan to show a demo of the solver happen in real time and graphs of the comparison on what happens when we parallelize the solution finder versus when we use a single thread to show a successful speedup.

PLATFORM:
We're choosing to use C++ so we can use OpenMP more easily. We also have quite a bit of experience coding in C++ so changing languages would likely increase our workload.

SCHEDULE:
November 9th - November 16th: Get a basic Perfect clear solver working.

November 16th - November 23rd: Add an OpenMP implementation to it and profile it.
November 23rd - November 30th: Add Tetris rules to our implementations and parallelize it.
November 30th - December 7th: Add Tetris rules to our implementations and parallelize it.
December 7th - December 14th: Try to optimize our algorithm.
December 14th - December 18th: Work on the final report and presentation