Project Milestone Report

Anupam Pokharel and Lisa Mishra

November 2021

1 Current Progress and Alignment with Proposed Schedule

1.1 Are We On-Schedule

We feel generally that our pacing is not very discrepant from what we intended at the proposal time, though there is some minor lag between our actual output of project progress and what was stated in the proposal. This is mostly on account of the greater intensity of the researching phase of the project (which involves reading and understanding the implementation mechanics of the API that allows porting of custom C++ routines into PyTorch and making those routines compatible with the inference-time abstractions that PyTorch already provides) than we anticipated. In response to this slight slowdown, we will operate by the following adjusted schedule, which differs from the proposal-time schedule mainly by omitting the "hope-for" goal for which there increasingly seems there will not be enough time.

1.2 Detailed Scheduled for the Remaining Time

WEEK OF	TO-DO ITEMS [NAME]
11-21 (1)	- Write milestone report [Both] - Complete OpenMP Implementation (of RNN Forward Pass) [Both]
11-21 (2)	- Obtain data and make graphs comparing speedups for various number of processors for CUDA Implementation [Anup]
11-28 (1)	- Obtain data and make graphs comparing speedups for various number of processors for OpenMP Implementation [Lisa]
11-28 (2)	- Expand CUDA implementation to work on higher processor counts (mainly for data collection on PSC) and, time permitting, more involved RNN inference computations (and gather the data) [Anup] - Replicate the expansions on the CUDA Implementation with that using OpenMP (including gathering the data) [Lisa]
12-5 (1)	- Graphs on data gathered from expansions for CUDA Implementation [Anup] - Graphs on data gathered from expansions for OpenMP Implementation [Lisa] - Polish results for demo [Both]
12-5 (2)	- Work on elements of our poster board, including graphs that demonstrate speedup for both CUDA and OpenMP implementations, and data analysis [Both]

1.3 Work Completed So Far

After having to invest more time than initially anticipated in researching the C++ API for PyTorch, we have completed the inference time forward-pass (depicted in the "BACKGROUND" of our proposal and our webpage) implementation in CUDA with kernel launches. This will enable us to collect data on the performance of the code with different processor counts and eventually to juxtapose the speedups with those obtained with an OpenMP-based implementation. Furthermore, we have acquired some valuable intuitions for where we can parallelize the OpenMP implementation (for example, we have a better understanding of spots in code where atomic operations would be apt).

2 Self-examining Progress with Respect to Project Goals

As we have alluded in the previous sections, the main change we are making upon having spent more time than initially allocated for researching is the omission of the “nice-to-have” goal of additionally composing an implementation of the RNN forward pass with OpenMPI (in order to complement our results for CPU-based speedups compared to the GPU-based speedups of the implementation with CUDA kernels). One reason we are omitting the OpenMPI-based implementation is our realization that increasing the amount and variety of data we gather for the OpenMP implementation will yield more-than-adequately substantive results. So, we plan to increase the breadth of the data-collecting phase of our project.

3 Plans for Poster Presentation

We intend on having graphs prepared that demonstrate the relative speedup between different number of processors for both our OpenMP and CUDA implementations. We would also like to have paragraphs for each that contain detailed analyses that describe why we see certain patterns regarding speedup (for example, why one implementation yields better performance results than the other) and to comment on any deviations from expected patterns. Additionally, we plan to have some content that summarizes and highlights the most important aspects of the analyses on the results, which we expect will be of interest to our classmates during the poster presentation.

4 Most Concerning Items

One thing we are concerned about is expanding both our OpenMP and CUDA implementations to include data for higher processor counts. Right now, the highest processor count we can have is 16 processors, but we would like to obtain more reliable and accurate results by obtaining the data with 32, 64, and 128 processors. This would require the usage of a machine like PSC (used for Assignments 3 and 4), and we are unsure about the amount of compute that is still available to us.