COL730 Assignment 5: Sorting with CUDA

Problem statement:

You are given an array consisting of 4-byte long unsigned integers. Design and implement a bitonic sorting network (also known as bitonic mergesort algorithm) using CUDA framework. The order used to sort is defined as follows:

$$A[i] \leq A[j] \; if \; and \; only \; if \; \sum_{p=0}^{i} A[p] \leq \sum_{q=0}^{j} A[q] \; modulo \; 2^{32} - 1$$

where,

A is the original array and A[i] denote its i^{th} element (assuming 0-based indexing).

Implement the function **sort()**. The function is called on the host with it's data pointer pointing to host memory and should satisfy the ordering condition as defined above on exit. It is expected to allocate and deallocate GPU memory, copy data between host and device as needed and launch the device kernel, finally updating the existing host buffer with the sorted data. The number of elements is guaranteed to be no more than 2^{30} elements. You are expected to exploit the CUDA shared memory hierarchy to optimize for execution time. A reference CUDA-based implementation of bitonic sort on 4-byte long unsigned integers is provided in file "sortcu_ref.cu". Note that it uses the integer order as the sort order instead of the order defined above. You are allowed to use and modify the provided implementation to fit the specification asked.

Submission instructions:

Submit a single zip file named [Your Entry Number].zip on moodle with the following:

1. An outline of the designed algorithm and its implementation choices made.
2. Table of execution time vs. size of input array.
3. Sources implementing the function signature provided in header "sortcu.h".
4. A makefile that builds a static library named "libsortcu.a".

Check course webpage for last date of submission.

Notes:

A significant weightage will be given to performance of sort() function.

You are expected to write your own driver program for testing and documentation as necessary, but the driver need not be included in your submission.