CME 213 Homework 3.

Problem 1.2

A screenshot of a cell phone

Description automatically generated

Problem 1.3

The bottleneck for char is mostly in the number of bytes we take in each instruction. In particular, we only take one byte each read for char but take four bytes each read for int. Hence, if we assume that the memory bus is not saturated, the speed up will be significant since we take four times more data in each read. However, when we increase the size of words from four bytes to eight bytes, as described in CUDA documentation, each memory request for a warp will now be split into two memory requests for each half-warp. This actually increase the number of times for reads so the bottleneck now becomes the memory throughput instead of instruction throughput, which is the case for char.

Problem 2.2

The total number of reads and writes will be equal to number of nodes times number of iterations times the total number of reads and writes in each loop. Note that in each loop we read graph\_indices twice, graph\_inv, graph\_edges and graph\_input number of times equal to the average number of edges, read number of nodes once and then write to the output once. Hence in each loop the read and writes is equal to (2\*sizeof(uint)+2\*edges\*sizeof(float)+edges\*sizeof(uint)+sizeof(float)+sizeof(uint)). Then the total size of reads and writes will be nodes\*iterations\*((edges+3)\*sizeof(uint)+(2\*edges+1)\*sizeof(float)).

Problem 2.3

Problem 2.4

The memory access is better in the middle of the table. This is because the kernel here is computationally more expensive, and thus the takes much more bandwith as number of nodes and average number of egdes increases.