Homework 4-1
Blocked All-Pairs Shortest Path Optimization

Parallel Programming 2020

Configuration

- Larger blocking factor
- Share memory in each block is 49152 byte.
  - 49152 byte = 12288 * sizeof(int) = 3 * 64 * 64 * sizeof(int)
  - So we choose 64 as our blocking factor.
- Padding to the multiple of 64
- round = v_padded / 64
- CUDA block dim: 32*32 (since there are 1024 threads in each block)
- CUDA grid dim: round * round

Share memory

- Copy the data to share memory before calculation
  - For phase 1: copy the pivot block to share memory
  - For phase 2: copy the pivot block and the calculating block to share memory.
  - For phase 3: copy the corresponding row block and column block to share memory.
- Each thread responsible for coping four entry in a block.

Branch

- Branch in kernel function cause warp divergence
- Reduce branch in cuda kernel.
- Use CUDA math API `__device__ int min(int a, int b)` instead of using `if else`.
- Unroll

2D Memory

- Use cudaMallocPitch and cudaMemcpy2D
- cudaMallocPitch may pad the allocation to ensure that corresponding pointers in any given row will continue to meet the alignment requirements for coalescing as the address is updated from row to row.

Others

- In phase 3, there is no need of calling `__syncthreads()` in each iteration. Only call it after all threads finish copying data from global memory to share memory.
- I/O acceleration.

Experiment

- Integer GOPS
  - Use nvprof --metrics inst_integer to get total number of integer instruction.
  - Divide by execution time.
- Global / Shared memory throughput.
  - Use flag gld_throughput, gst_throughput, shared_load_throughput and shared_store_throughput.