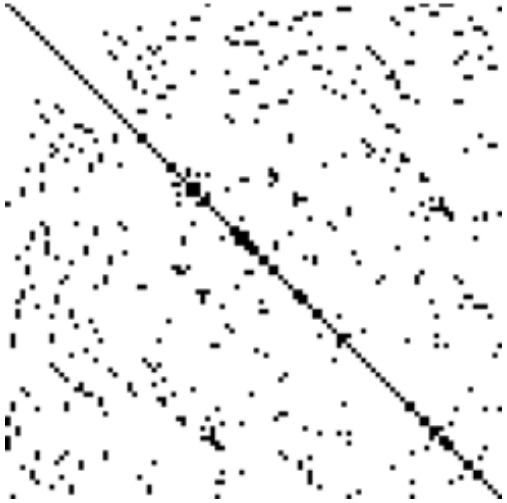
Improving performance for Matrix Multiplication in CUDA

We started off with Sparse Matrices



- Various representation formats of Sparse Matrices – COO, CSR, CSC, ELLPACK.
- Sequential algorithm for COO & CSR

Switched from Sparse to Dense Matrices

- For the CUDA implementation of Sparse Matrix multiplication, we started with simple dense matrix multiplication.
- Main task : Understand and compare Throughput oriented design (GPU) and Latency oriented design (CPU).
- Stuck to Matrix multiplication as the application – utilize the availability of SIMD.

Understanding the impact on performance

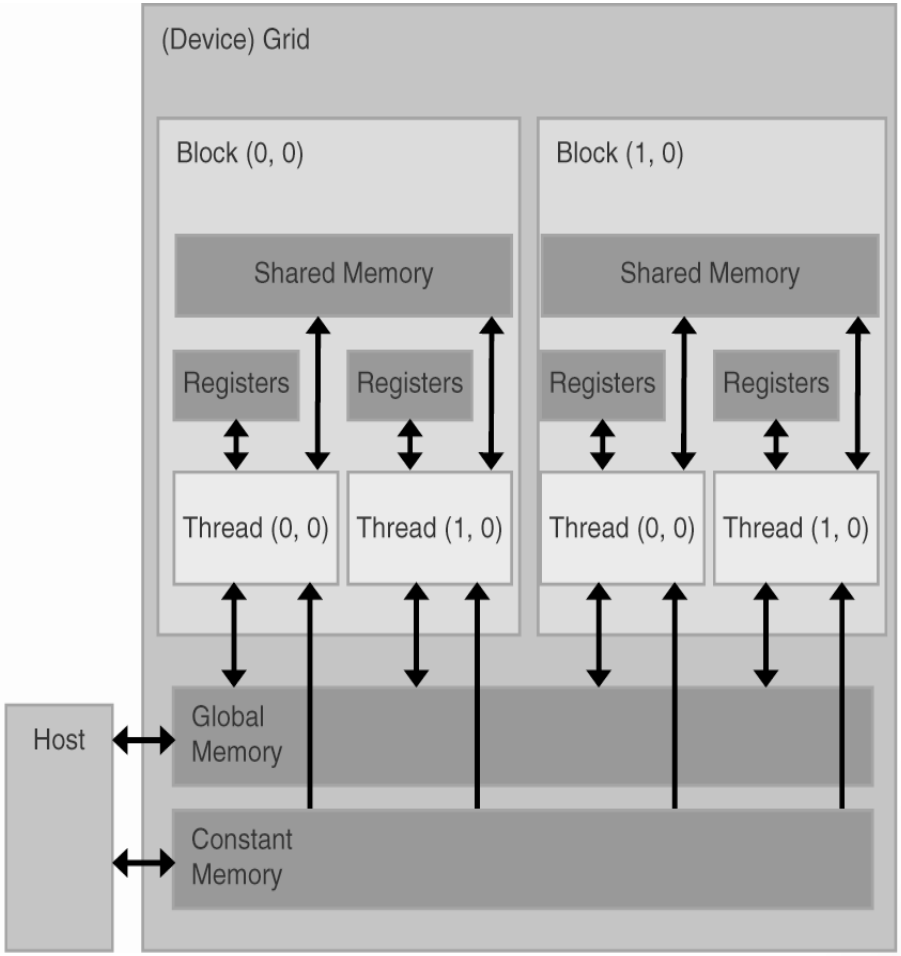
- Understanding the thread organization for CUDA
- CUDA Memories
- Use of GPU and CPU together to further enhance the performance.

Thread Organization

- Utilizing all the available threads in Streaming Multiprocessor of FERMI
  - Max number of resident threads per multiprocessor – 1536
  - Max number of blocks per multiprocessor – 8
  - Max number of threads per block – 512
- Identified the optimum number of threads per block and blocks per grid.
  - 16*16 is the optimum number of threads per block to achieve max performance.

CUDA Memories

- Device code can:
  - R/W per-thread registers
  - R/W per-thread local memory
  - R/W per-block shared memory
  - R/W per-grid global memory
  - Read only per-grid constant memory
- Host code can
  - Transfer data to/from per-grid global and constant memories

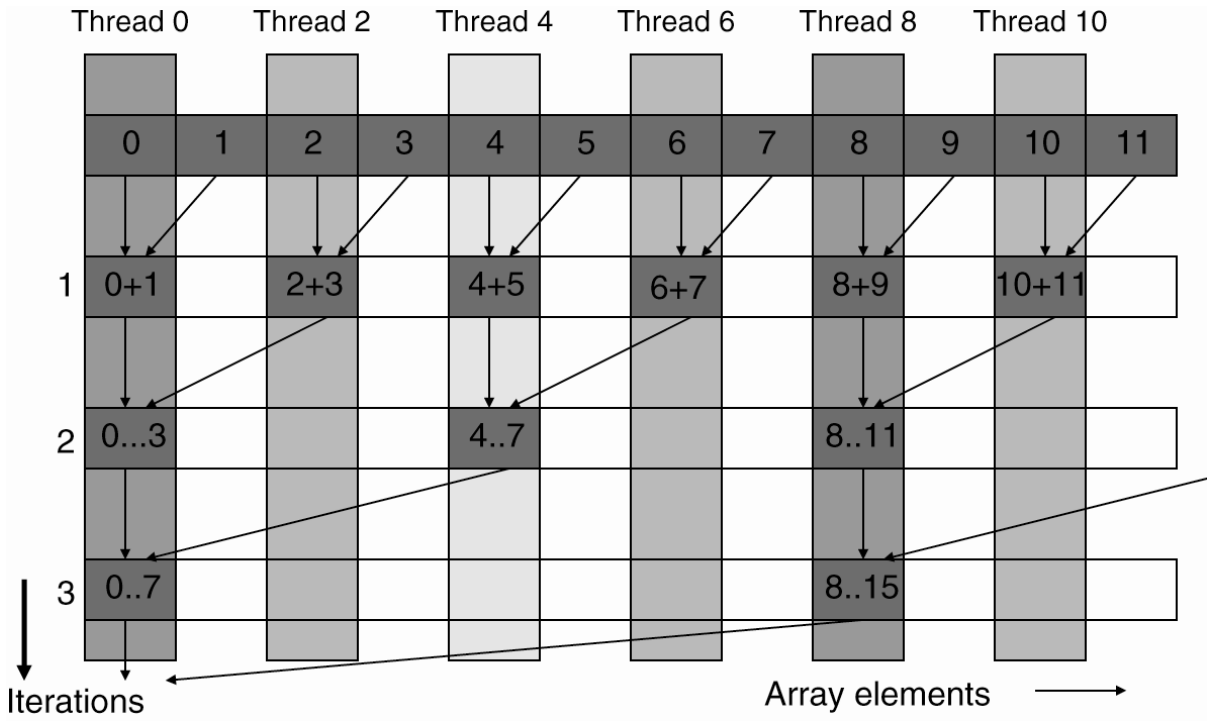


CUDA Memories

- Global memory resides in device memory(DRAM) which is much slower to access than shared memory
- Tiling the data, is a way to achieve higher performance to take advantage of shared memory.
- Access the Global memory once and save it in the shared memory and utilize the same.

Reduction

- A reduction algorithm extracts a single value from an array of values.



CPU and GPU together

- Can CPU perform reduction better than GPU?
- Moving the vector multiplication result from GPU to CPU and perform reduction on CPU.

#All images taken from Programming Massively Parallel Processors