HPCA Assignment - 03 Part-II

Name-kunal sah
Sr. number - 18042

Implementations:

1.Using cuda programming the gpu thread is implemented. Here we took thread block size = 16 because The number of threads in a thread block was formerly limited by the architecture to a total of 512 threads per block, but as of July 2019, with CUDA toolkit 10 and recent devices including Volta, blocks may contain up to 1024 threads.

2. Here total number of thread = 2*N -1 which is the same length of output pointer.

3. Each thread is calculating the value of each output, by using a loop which sums the products of diagonal values of matA and matB .

4. Using cudaMemcpyAsync the values of matA and matB is copied in device memory and using the same the output values is copied from device memory to host memory.

5. Here 1 D grid is used .

6. Here each gpu threads are not sharing any memory ,so they can run in parallel ,so here 2*N-1 threads are executed in parallel which reduces the runtime. In this way the gpu thread is optimized.

7. Further improvement can be done if the gpu thread can write in shared memory. Using these concept N*N number of threads will be running parallel and each thread will write in output pointer which will contain 2*N-1 number of values , for which multiple gpu threads will write to the same memory location , Using that further can be optimized.

Results:

For Input matrix of size 4096:

Reference execution time: 200.663 ms
GPU execution time: 98.62 ms
Speedup: 2.03471

For Input matrix of size 8192:

Reference execution time: 866.519 ms
GPU execution time: 198.043 ms
Speedup: 4.37541

For Input matrix of size 16384:

Reference execution time: 4515.78 ms

```
GPU execution time: 594.54 ms
Speedup: 7.59542
```