ASSIGNMENT HPC-2

Roll No: 41205

Problem Statement:

Design & implementation of Parallel (CUDA) algorithm to Add two large Vector, Multiply Vector and Matrix and Multiply two N × N arrays using n^2.

Objective:

1. To perform matrix operations using CUDA
2. Compare performance of algorithms sequentially and parallelly

Outcome: One will be able to write and compare more complex algorithms such as matrix multiplication parallelly.

Pre-requisites:

1. 64-bit Linux OS
2. Programming Languages: C/C++

Hardware Specification:

1. x86_64 bit
2. 2/4 GB DDR RAM
3. 80 - 500 GB SATA HD
4. 1GB NIDIA TITAN X Graphics Card

Software Specification:

1. Ubuntu 14.04
2. GPU Driver 352.68
3. CUDA Toolkit 8.0
4. CUDNN Library v5.0

Theory:

- Operations such as addition and multiplication of matrices take at least quadratic time, cubic for multiplication.
- This is feasible for smaller sized matrices, but the time is too high for matrices of large sizes such as 1000x1000.
- Since the operations of indices are independent of each other, these operations can be performed parallelly.
- Unlike vectors where blocks are used, we can use grid systems for matrix operations as they have a structure similar to matrices.

Syntax:

```
__global__ void function(args) {
    int i = blockIdx.x;
    int j = blockIdx.y;
    int idx = i * M + j;
    // operations
}

dim3 grid(N, M);
function<<<grid, 1>>>(args);
```

Test Cases:

#	Input	Expected Output	Actual Output	Result
1	Add to matrices	Sum calculated properly GPU faster than CPU	Sum calculated properly CPU: 905 microseconds GPU: 145 microseconds	Success
2	Multiply two matrices	Product calculated properly GPU faster than CPU	Product calculated properly CPU: 1205479 microseconds GPU: 145 microseconds	Success

Output:

Matrix Addition

```
MATRIX ADDITION:

CPU STATISTICS:
Time taken: 905 microseconds
GPU STATISTICS:
Time taken: 145 microseconds
```

Matrix Multiplication

```
MATRIX MULTIPLICATION:

CPU STATISTICS:
Time taken: 1205479 microseconds
GPU STATISTICS:
Time taken: 202 microseconds
```

Conclusion: Thus, we have performed matrix addition and multiplication operations parallelly using CUDA