Date:-

Assignment No - A2

Title:- Parallel Computing using CUDA.

Problem Statement:-
Vector and Matrix operations : Design parallel algorithm to :-
1) Add two large vectors.
2) Multiply vector and matrix.
3) Multiply two NxN arrays using n^2 process.

Learning Objectives:-
Learn parallel computing using CUDA, and parallel decomposition of a problem.

Outcome:-
Decomposed problem into sub-problems, learned how to use GPUs, solved sub problems using threads on GPU cores.

Software and Hardware Requirements:-
64 bit CPU, 4 GB RAM, Ubuntu / MacOS, CUDA toolkit, Nvidia GPU, NVCC compiler, Google colaboratory.

Theory:-
Dividing a computation into smaller computations and assigning them to different

processors for parallel execution are two key steps in the design of parallel algos.

The process of dividing a computation in smaller parts some or all which may potentially be executed in parallel is called decomposition.

Tasks are programmer defined units of computation into which the main computation is subdivided by means of decomposition.

Simultaneous execution of multiple tasks is the key reducing time required to solve the entire problem.

Tasks can be of arbitrary side, but once defined they are regarded as indivisible units of computation.

The tasks into which a problem is decomposed may not all be the same size.

1) In addition of two vectors, we have to add i^{th} element from first array with i^{th} element of second array to get i^{th} element of resultant array.
This allocation can be allocated to a distinct thread.
The same thing is done for vector product matrix.

Using CUDA vectors can be added using:-
1) n blocks, 1 thread/block.
2) 1 block, n threads.
3) m blocks, n threads/block.

2) Similarly, the product of a vector (1 x m) and matrix (m x n) will result in a 1 x n vector containing the result of the multiplication.

3) The product of two matrices ($m_1 \times n_1$) ($m_2 \times n_2$) will result in a matrix of dimension $n_1 \times n_2$.

CUDA Kernel and Threads.

The fundamental part of a CUDA code is the Kernel program.

Kernel is the function that can be executed in parallel in the GPU device.

A cuda Kernel is executed by an array of CUDA threads. All threads run the same code.

Each thread has an id that it uses to compute memory addresses and make control decisions CUDA organizes thousands of threads into a heirarchy of a grid of thread blocks.

A grid is a set of thread blocks that can be processed on a device in parallel.

A thread block is a set concurrent threads that can be cooperate among themselves through synchronization, barriers and access to a shared memory space private to the block.

Each thread is given an ID unique within the block; each block has a unique ID within the triad.

Conclusion:-

Successfully implemented and executed vector and matrix operations parallel using CUDA.