Assignment HPC-2

Title: Vector and operations using CUDA

Problem Statement
Design parallel algorithm to
1.) Add two large vectors
2.) Multiply vector and matrix
3.) Multiply two NxN arrays n^2 processor

Objectives
To learn CUDA architecture and programming language

Outcomes
To be able to learn CUDA architecture and programming concepts

Software and hardware requirements
OS: Fedora / Windows 10 (64-bit)
CUDA API, NVCC compiler, NVIDIA GPU / Google Colab
RAM: 4 GB, HDD: 500 GB

Concepts related theory
CUDA architecture
The architecture consists of several components like
1) Parallel compute engines inside NVIDIA GPUs
2) OS-kernel level support
3) User-mode driver providing device level-API
4) PTX instruction set architecture for parallel computing kernels and functions

CUDA memory hierarchy
- Each thread has private local memory
- Each Thread block has shared memory visible to all threads in block
- All Threads have access to global memory.

Compilation with NVCC - compiler driver to simply process compiling of C++ programs

Algorithms:

1. Addition
In addition of vectors, add i^{th} element from first array to i^{th} of second. Each array addition can be done in different thread. Cases -
i) n blocks and 1 thread per block
→ id = blockIdx.x
add <<< n, 1 >>> ();
ii) 1 block and n threads per block
→ id = threadIdx.x
add <<< 1, n >>> ();
iii) m blocks and n threads per block
id = blockIdx.x * blockDim.x + threadIdx.x

2. Multiplication
i) 2D blocks and one thread per block
Here, x = blockIdx.x
y = blockIdx.y
grid dimensions ⇒ dim3 grid (col, row)
Function kernel ⇒ matproduct <<< grid, 1 >>> (l, m, n);

Speedup = $\frac{\text{time serial}}{\text{time parallel}}$

Matrix linearisation in CUDA

row = blockIdx.y * blockDim.y + threadIdx.y

col = blockIdx.x * blockDim.x + threadIdx.x

offset = row * N + column

Test Cases

	Function	Size	Serial time (ms)	Parallel time (ms)	Speed up
1.	Vector addition	a) 10000	0.0475	0.059	0.8
		b) 50000	0.0195	0.145	1.34
2.	Vector - matrix multiplication	a) 1*10 and 100*100	0.059	0.053	1.11
3.	Matrix - matrix multiply	a) N=10	0.006	0.022	0.257
		b) N=50	0.057	0.074	0.82

Conclusion

Thus we successfully implemented vector and matrix operation using parallel computing and CUDA. Analysed the speedup for various operations. For small arrays speedup is less than 1, but as the size increases parallel computing becomes less costly.