LP-1

High Performance Computing

Assignment 1.

Date of Completion – 19 · 8 · 2020

Title :- Parallel Reduction using CUDA

Problem Statement :-

a) Implement parallel reduction using MIN, MAX, SUM & AVG operation

b) Write a CUDA program that, given an N-element vector find
- The maximum element in the vector
- The minimum element in the vector
- The arithmetic mean of the vector
- The standard deviation of the values in the vect

Test for input N and generate a randomized vector V of length N (N should be large) The program should generate sup output as the two compared maximum values as well as the time taken to find each value.

Objectives :- a) To learn parallel programming concepts.
b) To learn parallel computing using CUDA.

Outcomes :- a) Know parallel computing concepts
b) Use CUDA for parallel programming.

Requirements :- Ubuntu OS, Nvidia GPU, CUDA API (c/c++) Google Colab

Theory :- CUDA :-

1) It is a parallel Computing platform and API model created by NVIDIA.
2) It enables programmers to use CUDA enable GPU for general purpose processing
3) It is a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements, for the execution of Computer kernels.

CUDA Programming

1) NVCC compiler separates the host and device code (GPU) in compilation phase.
2) Source code has .cu extension.

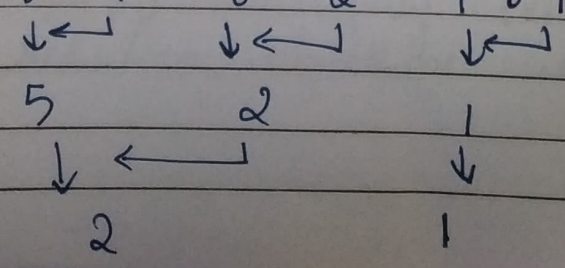
Program Structure:-

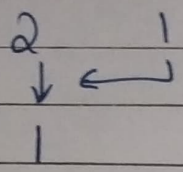
1) include headers
2) allocate GPU memories
3) Copy data from CPU to GPU.
4) Invoke the Kernel code
5) Copy data back from GPU to CPU
6) Destroy GPU memories.

Parallel Reduction:-

1) Here Every thread calculate result from its own element and some other element.
2) This result is forwarded to next round of threads.
3) Number of threads are halfed in each round untill single thread remains.

Eg for calculating minimum from 6 element array





Test case :-

for n = 1000

minimum = ~~121~~ 3

parallel execution time : 8ms

normal execution time : 24ms

4 maximum = 199

parallel execution : 2ms

normal execution : 38ms

Sum : 1210

Parallel execution : 2ms

normal execution : 15ms

Standard deviation : 0.11726

parallel execution : 2ms

Efficiency = $\frac{WSA}{WCPA}$

Worst case execution time for sequential algorithm => WCSA

Worst case ~~parallel~~ execution time for parallel algorithm => WCPA

Conclusion :- ~~we~~ I successfully executed the parallel reduction algorithms using CUDA.