Sudesh Pawar
BE Comp A
PRN: F17111037

HPC
A1 - 1

Questions:

1) How is CUDA programming useful to study parallel algorithms?

Ans i) CUDA enables developers to speed up compute-intensive applications by harnessing power of GPUs for parallelizable part of computation

ii) CUDA is ideal for an parallel problem, where little or no interthread or interblock communication is required.

iii) It supports inter thread communication with explicit primitives using on chip resources.

iv) Parallel portion of application is executed as CUDA kernel.

v) CUDA splits problems into grids of blocks, each containing multiple blocks which may run in any order.

vi) In this way, CUDA uses its resources fully to execute parallel programming.

vii) Hence, it is useful for studying parallel algorithms.

2) Discuss importance of parallel reduction operations

Ans. i) In parallel programming, main criterion is to have parallel threads in the program.

ii) In order to run sections in parallel, they shouldn't have a dependence relationship between them

iii) A reduction operation can help breakdown a task into various partial tasks.

iv) Due to parallel operations, performed the no. of steps required for certain operations are reduced

v) The smaller operations are later merged together ~~to~~ find the final result.

vi) Eg: For addition & multiplication of nos. in an array.

vii) Thus, reduction operation helps remove the data dependencies & allow multiple threads to run at same time thus parallelizing the task.

3) Discuss operations on vectors & the approach for parallel algorithm design for same.

Ans. i) Matrix computation is highly important in many standards software libraries contain procedures for various matrix operation.

ii) The amount of software for matrix programming /processing is constantly increasing new sufficient storage structure forspecial type matrix are being created.

iii) Highly efficient machine dependent algorithm implementation are being developed. The theoretical research information searching, faster matrix calculation method is being carried out.

Q.4) How is parallelism achieved in CUDA?

Ans. i) CPU recognises serial code & parallel code for execution.

ii) CPU thread is initialized to execute the serial code & GPU thread is initialized for parallel code.

iii) GPU thread copies data for parallel processing from main memory to GPU memory.

iv) CPU initiated GPU compute kernel & hands over the control to GPU.

v) GPU's CUDA cores execute kernel in parallel.

vi) CUDA splits problems into grids of blocks, each containing multiple threads

vii) The blocks may run in any order.

viii) Only a subset of blocks will ever execute at any one point in time.

ix) A block must execute from start to completion & may be run on one of NSMs (Symmetrical Multiprocessors)

x) Blocks are allocated from grid of blocks to any SM that has free slot.

xi) Initially this is done on round-robin basis so each SM gets an equal distribution of blocks.

xii) For most kernels, the no. of blocks need to be in the order of eight or more times the no. of physical SMs on GPU.

Q.5) Explain Grid, Block & thread structure in relation with parallel reduction.

Ans. • Grid: Grid is a group of blocks. There is no synchronization at all between these blocks.

• Blocks: Blocks are group of threads. Each thread can communicate within its own block.
There is nothing much you can say about, the execution concurrently or serially & also it has no particular order.

• Thread: Thread is just an execution of kernel with given index. Each thread uses its index to access element in array such that collection of all thread co-operate in the process for entire dataset.