An Even Easier Introduction to CUDA

Jeong Hoon Choi
USCID: 5023184813

DSCI-560
Data Science Professional Practicum

Introduction to CUDA

- __global__: kernel functions callable from the host
- __device__: kernel functions callable from the kernel (__device__ or __global__)
- __host__: host functions callable from the host

- cudaMallocManaged(void** dptr, size_t size, unsinged int flags=cudaMemAttachGlobal): allocates memory (manged by the unified memory system)
- cudaMalloc(void** dptr, size_t size): allocate memory on the device(GPU)
- cudaMemPrefetchAsync(const void* dptr, size_t count, cudaMemLocation, unsigned int, cudaStream_t): prefetch memory to the specified destination location
- cudaFree(void* dptr): frees memory on the device (GPU)

Introduction to CUDA

global function call:

```
function_name<<<gridDim, blockDim, sharedMemBytes, stream>>>(argument1, argument2);
```

cuda synchronize:

cudaDeviceSynchronize(): wait for compute device to finish

Introduction to CUDA

Cuda Kernel Built-in functions and variables:

__syncthreads(): all threads in the block sync

threadIdx.x, threadIdx.y, threadIdx.z

blockDim.x, blockDim.y, blockDim.z

(0 <= threadIdx.x < blockDim.x)

blockIdx.x, blockIdx.y blockIdx.z

warpSize: size of warp

Kernel Keywords:

__shared__: shared memory (block share memory)

Introduction to CUDA

cuda C++

__host__ → __global__ → __device__

malloc(host memory allocate)

↕

cudamalloc(device memory allocate)

cudaFree (device memory free)

bultin-variables (thread)
blockIdx , blockDim , threadIdx

cudaMemcpy
: cudaMemcpyDeviceToHost
: cudaMemcpyHostToDevice

call global function:
global_function<<<gridDim , blockDim>>>

Question: Explain function of CUDA memory hierarchy and its scope

Answer:

http://thebeardsage.com/cuda-memory-hierarchy/

Global Memory: all threads and CPU, cudaMalloc, cudaFree

Constant Memory: all threads and CPU (RO), __constant__ int const_data;

Shared Memory: per block, __shared__ int shared_data[64];

Local Memory: per thread, int local_data[64]; (decide by compiler)

Register: per thread, int register_data; (decide by compiler)