OpenMP and CUDA Summary

OpenMP (Open Multi-Processing)

Definition

OpenMP is an API that supports multi-platform shared-memory multiprocessing programming in C, C++, and Fortran on most architectures, including Unix and Windows.

Key Features

- Based on compiler directives (pragmas).
- Supports parallel for-loops, sections, tasks, synchronization, and reductions.
- Best suited for shared-memory systems.

Important Directives

Directive	Purpose
`#pragma omp parallel`	Creates a team of threads
`#pragma omp for`	Distributes loop iterations
`#pragma omp sections`	Executes code blocks in parallel
`#pragma omp critical`	Ensures mutual exclusion
`#pragma omp barrier`	Synchronization point
`#pragma omp reduction`	Performs parallel reduction operations

Advantages

- Easy to use (incrementally parallelize code).
- No need to manage threads manually.
- Efficient on multicore CPUs.

Limitations

- Only works for shared memory.
- Limited scalability compared to MPI or CUDA.

Viva Questions

Question	Answer
What is OpenMP?	A set of compiler directives for parallel programming on shared-memory systems.
How do you parallelize a loop in OpenMP?	Using `#pragma omp parallel for`.
What is the purpose of reduction in OpenMP?	To perform parallel aggregation (e.g., sum, max).
What are the main types of OpenMP constructs?	Parallel, work-sharing, synchronization, and data environment.
What is critical section in OpenMP?	A section that allows only one thread to execute it at a time.

CUDA (Compute Unified Device Architecture)

Definition

CUDA is a parallel computing platform and API developed by NVIDIA that allows programmers to use GPUs for general-purpose computing (GPGPU).

Key Concepts

- **Kernel**: A function that runs on the GPU.
- **Thread**: Basic unit of execution on the GPU.
- **Block**: A group of threads.
- **Grid**: A group of blocks.

CUDA Memory Types

Memory Type	Scope	Speed
Global Memory	Accessible by all threads	Slow
Shared Memory	Shared within a block	Fast
Local Memory	Private to a thread	Medium
Constant Memory	Read-only, all threads	Fast (cached)

Kernel Syntax

```
__global__ void add(int *a, int *b, int *c) {
    int idx = threadIdx.x;
    c[idx] = a[idx] + b[idx];
}
// Called from host:
add<<<1, n>>>(a, b, c);
```

Advantages

- Massive parallelism using thousands of GPU cores.
- High performance for data-parallel tasks like image processing, ML, etc.

Limitations

- Works only with NVIDIA GPUs.
- More complex than OpenMP.
- Requires managing memory between CPU and GPU.

Viva Questions

Question	**Answer**
What is CUDA?	A parallel computing platform to run code on NVIDIA GPUs.
What is a kernel in CUDA?	A function executed in parallel by GPU threads.
What are thread, block, and grid in CUDA?	Thread: smallest unit, Block: group of threads, Grid: group of blocks.
How is CUDA different from OpenMP?	CUDA uses GPUs (SIMD); OpenMP uses CPUs (multithreading on shared memory).
What is __global__ in CUDA?	Declares a kernel function that runs on the device (GPU) and is called from the host (CPU).
What types of memory are in CUDA?	Global, shared, local, constant.
What is coalesced memory access?	Aligned memory access that improves performance.
What is warp in CUDA?	A group of 32 threads executed together on GPU.