Introduction to GPUs
GPUs

• Graphics Processing Units (GPUs) are co-processors.
  ▪ A hardware that runs programs in addition to CPU.
  ▪ Originally, those programs displayed graphics.
  ▪ GPUs run programs with a lot of computations.

• CPU uses task parallelism while GPU uses data parallelism.
GPUs versus CPUs

• Task parallelism
  – Threads perform different computations on the same or different data.
  – An array $A$ of size $N$ ($A[0] \ldots A[N-1]$)
  – Thread 1 computes the sum of $A[0] \ldots A[N]$ , but Thread 2 computes the average of $A[0] \ldots A[N]$.

• Data parallelism
  – Threads perform the same computation on different parts of the same data.
  – E.g., the password cracker lab uses the data parallelism.

❖ Task parallelism distributes tasks. Data parallelism distributes data.
GPUs

• GPU is an array of streaming multiprocessors (SMs)
• A multithreaded program is divided into blocks of threads.
• Each block executes independent of the other blocks.
CUDA

• A programming model that uses data parallelism for speeding up program executions.
  – Used as an extension for a programming language

• Uses GPUs to execute the parallel portion of compute-intensive programs

• Originally built for graphics application, but later used for general-purpose programs
CUDA

- Programs developed by CUDA run on GPU and CPU.
  - **Host**: CPU
  - **Device**: GPU
  - CPU and GPU have their own memory spaces.

- Programs launch **kernels**
  - Kernels are functions executed on GPUs.
  - When launched, threads are generated.
  - A kernel data is divided and processed by many threads in parallel.
Thread Hierarchy

- Thread blocks are mapped to SMs.
- Thread blocks run in an order.
- Threads of a block run in parallel.

- The thread block is the basic unit of execution
Memory Hierarchy

Thread ← Per-thread *local* memory

Thread block ← Per-block *shared* memory

Grid

Block (0, 0)  Block (1, 0)

Block (0, 1)  Block (1, 1)

Global memory
Heterogeneous System

C program

Serial code

Parallel code (kernel)

Serial code
CUDA Memory
Host and Global Memory

• Host memory is the CPU memory.
• Device memory is the GPU **global** memory.
  – Holds dynamically allocated memory.
  – `cudaMalloc` returns a pointer to a memory block.
  – Pass the pointer to a kernel
  – `cudaMemcpy` transfers data between host and device.

![Diagram of host and device memory with CUDA functions](attachment:diagram.png)
Managed Memory

• Also called unified memory (abstraction)
  – CUDA transfers data between host and device memories using `cudaMallocManaged`
  – Replaces `cudaMalloc` and `cudaMemcpy()` routine

• Simpler codes, but not faster programs
Managed Memory

__global__ void AplusB(int *ret, int a, int b) {
    ret[threadIdx.x] = a + b + threadIdx.x;
}

int main() {
    int* ret;
    cudaMemcpy(ret, host_ret, 1000 * sizeof(int));
    AplusB<<<1, 1000>>>(ret, 10, 100);
    int* host_ret = (int*)malloc(1000 * sizeof(int));
    cudaMemcpy(host_ret, ret, 1000 * sizeof(int));
    for(int i=0; i<1000; i++) {
        printf("%d\n", host_ret[i]);
    }
    free(host_ret);
    cudaFree(ret);
    return 0;
}
Managed Memory

__global__ void AplusB(int *ret, int a, int b) {
    ret[threadIdx.x] = a + b + threadIdx.x;
}

int main() {
    int* ret;
    cudaMallocManaged(&ret, 1000 * sizeof(int));
    AplusB<<<1, 1000>>>(ret, 10, 100);
    cudaDeviceSynchronize();
    for(int i=0; i<1000; i++) {
        printf("%d\n", ret[i]);
    }
    cudaFree(ret);
    return 0;
}
Shared Memory

• Block threads use the shared memory to synchronize their executions.
• Faster memory (shared only by threads of a block)

• Example
  – Multiply two matrices A and B
  – width: Number of columns
  – height: Number of rows
  – \([A.\text{height}, A.\text{width}] \times [B.\text{height}, B.\text{width}]\)
  – \([A.\text{height}, B.\text{weight}]\)
Without Shared Memory

A is read B.width times.
B is read A.height times.
Sub-matrices are loaded from global memory to shared memory.

A is read $B\cdot \text{width}/\text{blocksize}$ times.
B is read $A\cdot \text{height}/\text{blocksize}$ times.