Hướng dẫn thực hành tuần 2

Bài toán đầu tiên trên CUDA

Mục đích: làm quen với việc lập trình trên CUDA thông qua những bài toán đơn giản nhất.

Nội dung:

- Trình bày một ví dụ cụ thể.
- Bài tập

I. Hướng dẫn:

Bài toán: cho một mảng một chiều, tăng các phần tử trong mảng này thêm 1 đơn vị.

Ví dụ nảy sẽ trình bày 2 cách thực hiện: bằng CPU (cách làm thông thường) và GPU (bằng CUDA).

1. Hàm tăng mảng trên host (CPU)

```
void IncrementArrayOnHost(float* a, int N)
{
    for (int i = 0;i<N;i++)
        a[i] = a[i] + 1.f;
}
```

2. Hàm tăng mảng trên device (GPU)

```
__global__ void IncrementArrayOnDevice(float* a, int N)
{
    int idx = blockIdx.x * blockDim.x + threadIdx.x;
    if (idx < N)
        a[idx] = a[idx] + 1.f;
}
```

❖ **Lưu ý:** hàm trên host có sử dụng vòng lặp còn hàm trên device thì không. Mỗi tiểu trình sẽ tương ứng với một phần tử trong mảng.

3. Hàm main:

❖ **Nhắc lại:** một chương trình CUDA chuẩn gồm 5 phần:

- Cấp phát bộ nhớ trên device

- Copy dữ liệu từ host sang device
- Gọi hàm thực thi trên device
- Copy kết quả từ device sang host
- Giải phóng bộ nhớ

Các bước thực hiện cụ thể như sau:

- Khai báo mảng trong host và device

```
float *a_h, *b_h;      // pointers to host memory
float *a_d;               // pointers to device memory
int N = 18;               // Size of array
int i;
```

- Cấp phát bộ nhớ trong host và device

```
// allocate arrays on host
a_h = (float *)malloc(sizeof(float)*N);
b_h = (float *)malloc(sizeof(float)*N);
// allocate arrays on device
cudaMalloc((void **) &a_d, sizeof(float)*N);
```

- Khởi tạo giá trị mảng:

```
// initialize host data
for (i=0; i<N; i++)
        a_h[i] = 10.f+i;
```

- Gọi hàm tăng mảng trong host

```
// Call host function
IncrementArrayOnHost(a_h,N);
```

- Sao chép mảng từ host sang device và gọi hàm tăng mảng trên device

```
// send data from host to device: a_h to a_d
cudaMemcpy(a_d, a_h, sizeof(float)*N,
cudaMemcpyHostToDevice);
// Call device function
int blockSize = 4;
int nBlocks = N/blockSize + (N% blockSize == 0 ? 0:1);
IncrementArrayOnDevice<<<nBlocks, blockSize>>> (a_d, N);
```

- Sao chép dữ liệu trở lại từ device sang host

```
   // retrieve data from device: b_d to b_h
   cudaMemcpy(b_h, a_d, sizeof(float)*N,
cudaMemcpyDeviceToHost);
```

- Kiểm tra kết quả và giải phóng tài nguyên.

```
 // check result
for (i=0; i<N; i++)
   printf("%f : %f \n",a_h[i],b_h[i]);
// cleanup
free(a_h);
free(b_h);
cudaFree(a_d);
```

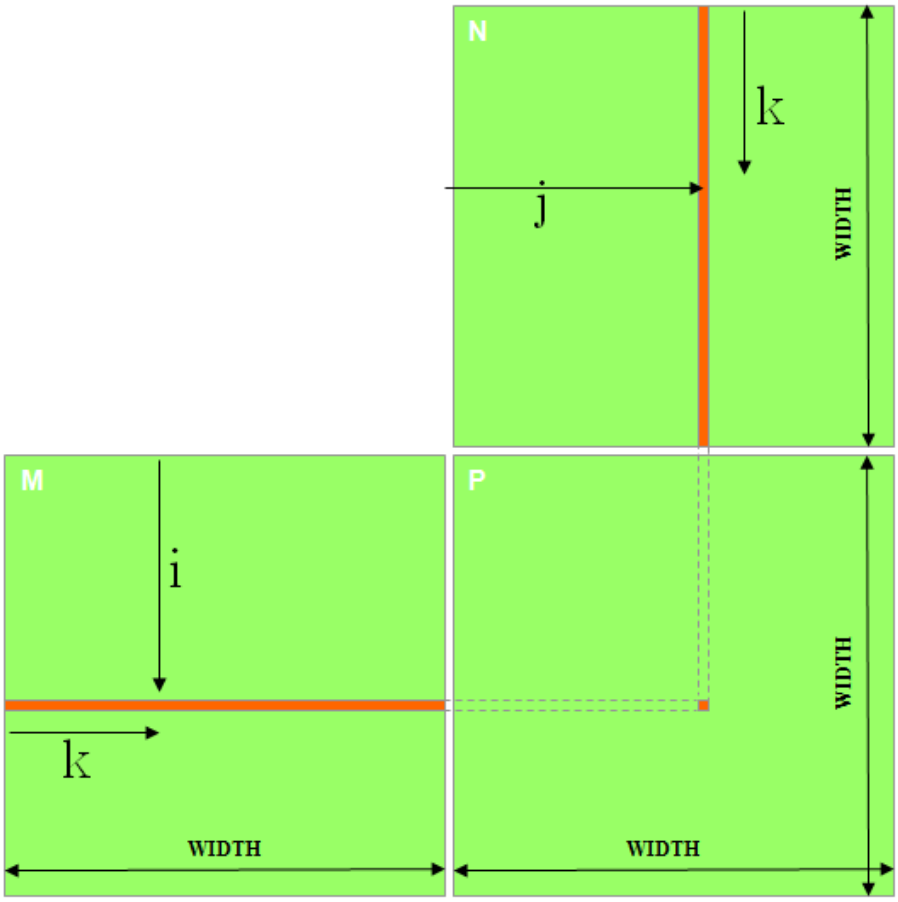
II. Bài tập:

1 . Đề bài: Viết chương trình thực hiện bài toán nhân hai ma trận **vuông** trên host và device.

2. Thời gian: 1 tuần (hạn chót 04/10/2009)

3. Cách nộp: nộp qua moodle.

Gợi ý:



- Biểu diễn ma trận (mảng hai chiều) bằng mảng một chiều.
- Mỗi tiểu trình sẽ tình một phần tử trong ma trận P.
- Để đơn giản, chỉ sử dụng 1 block với kích thước width * width

  dim3 dimGrid(1, 1);

  dim3 dimBlock(Width, Width);