Lập trình song song trên GPU

Hoàng Minh Thanh (18424062)

TH1 : Làm mờ ảnh

Bộ môn Công nghệ phần mềm

Khoa Công nghệ thông tin

Đại học Khoa học tự nhiên TP HCM

Contents

I. Quá trình cài đặt 3

1) Cài đặt hàm blurImg trên GPU 3

2) Khởi tạo và thực thi trên GPU 4

II. Kết quả thực thi 5

1) Biên dịch 5

2) 8x8 6

3) 16x16 7

4) 32x32 7

5) 64x64 8

Giải thích : 8

Vì máy tính cả nhân của em không có GPU nên bắt buộc em phải sử dụng Google Colab :
https://colab.research.google.com/drive/189f2vZtNKGjc5oc5T6dkOJOVClT4rnS1

(Thầy có thể vào link Online để xem luồng chạy dễ hơn báo cáo)

I. Quá trình cài đặt

Chi tiết cài đặt trong file .cu và file Google Colab

Block và Grid được cấu hình theo **2D grid** và **2D block**

Vì các file thực thi và câu lệnh em để trên github private repos nên phải cấu hình clone bằng ssh trên Google Colab

1) Cài đặt hàm blurImg trên GPU

Cài đặt :

Với mỗi thread trong GPU ta sẽ lấy ra ma trận filter và nhân với kết quả pixel input của các pixels kế cận

Sau đó ta chỉ cần gán lại kết quả vào pixelOutput đầu ra

Chú ý kiểm tra kích thước của thread

```cpp
__global__ void blurImgKernel(uchar3 * inPixels, int width, int height,
                              float * filter, int filterWidth,
                              uchar3 * outPixels){
    // TODO
    int r = blockIdx.y * blockDim.y + threadIdx.y;
    int c = blockIdx.x * blockDim.x + threadIdx.x;
    if (r < height && c < width)
    {
        int i = r * width + c;
        float3 outPixel = make_float3(0, 0, 0);        You, 28 minutes ago •
        for (int fR = 0; fR < filterWidth; fR++)
        {
            for (int fC = 0; fC < filterWidth; fC++)
            {
                float filterVal = filter[fR * filterWidth + fC];

                int inPixelsR = (r - filterWidth/2) + fR;
                int inPixelsC = (c - filterWidth/2) + fC;
                inPixelsR = min(height - 1, max(0, inPixelsR));
                inPixelsC = min(width - 1, max(0, inPixelsC));
                uchar3 inPixel = inPixels[inPixelsR * width + inPixelsC];

                outPixel.x += filterVal * inPixel.x;
                outPixel.y += filterVal * inPixel.y;
                outPixel.z += filterVal * inPixel.z;
            }
        }

        outPixels[i] = make_uchar3(outPixel.x, outPixel.y, outPixel.z);
    }
}
```

2) Khởi tạo và thực thi trên GPU

Đầu tiên ta cần khởi tạo bộ nhớ của pixel trên GPU, và cũng như filer

Sau đó ta copy dữ liệu từ pixels input vào các biến trên GPU

Và sau đó gọi lệnh thực thi trên block vừa tính được

Sau khi thực thì xong thì copy kết quả đã tính toán từ GPU ra và hủy bộ nhớ

```
// TODO
// Allocate device memories
uchar3 *d_inPixels, *d_outPixels;
float *d_filter;
CHECK(cudaMalloc(&d_inPixels, width * height * 3 * sizeof(uchar3)));
CHECK(cudaMalloc(&d_outPixels, width * height * 3 * sizeof(uchar3)));
CHECK(cudaMalloc(&d_filter, filterWidth * filterWidth * sizeof(float)));

// Copy data to device memory
CHECK(cudaMemcpy(d_inPixels, inPixels, width * height * 3 * sizeof(uchar3), cudaMemcpyHostToDevice));
CHECK(cudaMemcpy(d_filter, filter, filterWidth * filterWidth * sizeof(float), cudaMemcpyHostToDevice));

// Call kernel
dim3 gridSize((width - 1) / blockSize.x + 1, (height - 1) / blockSize.y + 1);
blurImgKernel<<<gridSize, blockSize>>>(d_inPixels, width, height, d_filter, filterWidth, d_outPixels);

// Copy result from device memory
CHECK(cudaMemcpy(outPixels, d_outPixels, width * height * sizeof(uchar3), cudaMemcpyDeviceToHost));

// Free device memories
CHECK(cudaFree(d_inPixels));      You, 34 minutes ago • update TH1
CHECK(cudaFree(d_outPixels));
CHECK(cudaFree(d_filter));
```

II. Kết quả thực thi

1) Biên dịch

- Biên dịch file "bt1.cu"

```
[10]   1 !nvcc bt1.cu -o bt1
```

Ta thực thi với cấu hình mặc định

```
[18]  1 !./bt1 in.pnm out.pnm
```

```
Image size (width x height): 512 x 512

Processing time (use host): 386.711151 ms

GPU name: Tesla P100-PCIE-16GB
GPU compute capability: 6.0
Processing time (use device): 2.148000 ms

Error between device result and host result: 0.000703
```

2) 8x8

```
[14]  1 !./bt1 in.pnm out.pnm 8 8
```

```
Image size (width x height): 512 x 512

Processing time (use host): 394.060913 ms

GPU name: Tesla P100-PCIE-16GB
GPU compute capability: 6.0
Processing time (use device): 2.206176 ms

Error between device result and host result: 0.000703
```

3) 16x16

```
[15]  1 !./bt1 in.pnm out.pnm 16 16
```

```
Image size (width x height): 512 x 512

Processing time (use host): 393.984985 ms

GPU name: Tesla P100-PCIE-16GB
GPU compute capability: 6.0
Processing time (use device): 2.061920 ms

Error between device result and host result: 0.000703
```

4) 32x32

```
[16]  1 !./bt1 in.pnm out.pnm 32 32
```

```
Image size (width x height): 512 x 512

Processing time (use host): 385.749817 ms

GPU name: Tesla P100-PCIE-16GB
GPU compute capability: 6.0
Processing time (use device): 2.049952 ms

Error between device result and host result: 0.000703
```

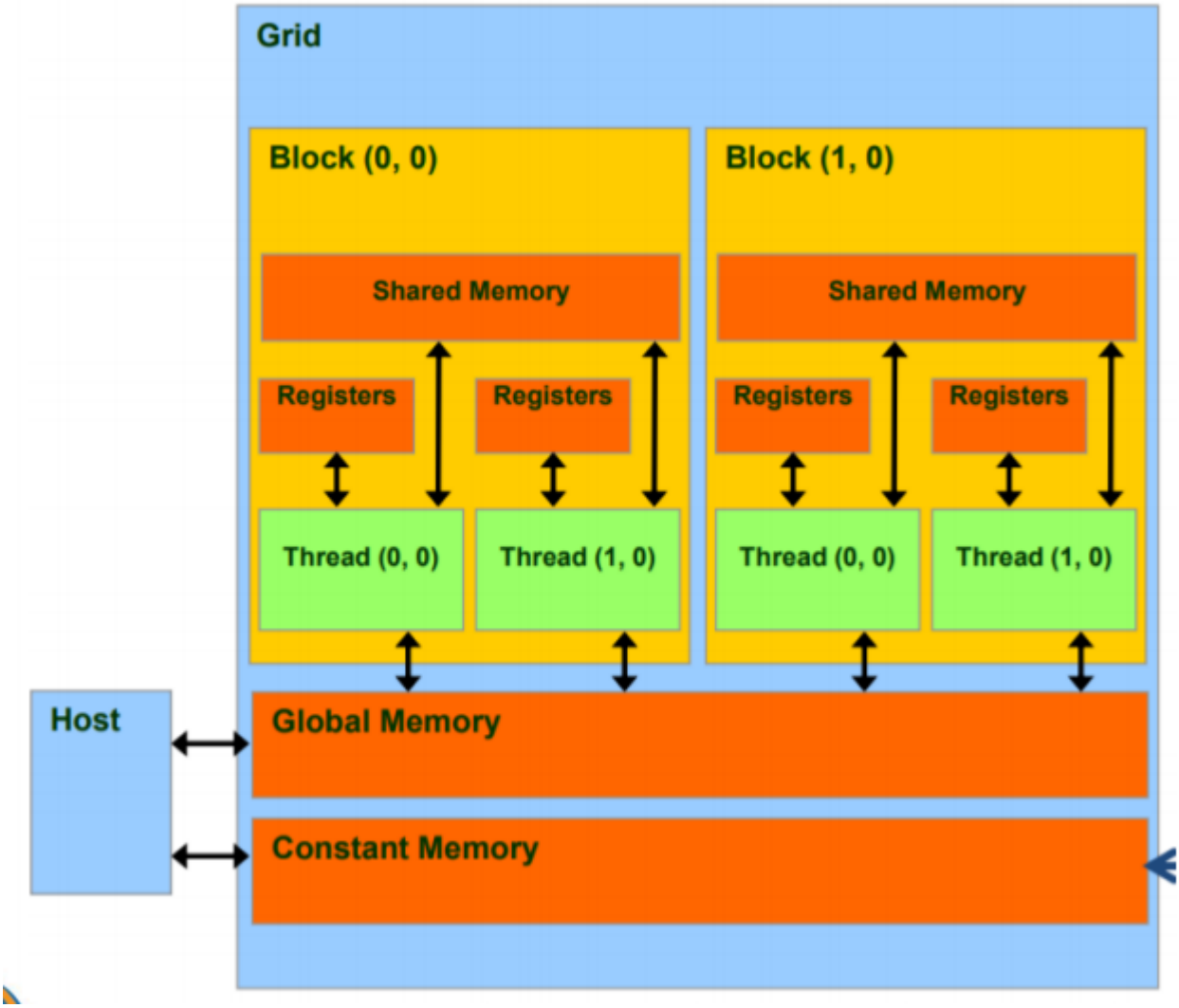
5) 64x64

```
1 !./bt1 in.pnm out.pnm 64 64

Image size (width x height): 512 x 512

Processing time (use host): 387.610596 ms

GPU name: Tesla P100-PCIE-16GB
GPU compute capability: 6.0
Processing time (use device): 1.925440 ms

Error between device result and host result: 124.033791
```

Có thể thấy độ lỗi lên đến **124.033791** khi thực thi với kích thước filter 64 x 64 đúng như với yêu cầu của thầy

Giải thích :

Xảy ra lỗi như vậy là có thể là do các thread của mỗi block được phân vào các wrap (32 thread) khác nhau. Trong GPU thì mỗi thread trong một wrap thực thi bất đồng bộ sử dụng chung một thanh nhớ register (bộ nhớ tạm trên cùng một wrap) nên có thể với kích thước filter lớn hơn 32 thì quá trình tính toán song song sẽ dẫn đến một thread đã xử lý xong và cập nhật kết quả mới vào bộ nhớ toàn cục nhưng thread khác đang xử lý sẽ lấy kết quả pixels từ bộ nhớ tạm trên thanh ghi.


Grid
Block (0, 0)
Block (1, 0)
Shared Memory
Shared Memory
Registers
Registers
Registers
Registers
Thread (0, 0)
Thread (1, 0)
Thread (0, 0)
Thread (1, 0)
Host
Global Memory
Constant Memory