ECE 786: Advanced Computer Architecture: Data Parallel Processors

Programming Assignment 1 Report - 2D Convolution

-Shantanu Mukhopadhyay[200541793]

Part A: Code Explanation:

Host Code Workflow

The host code follows these main steps:

1. **Reading Input and Filter Files:**
   - The readInputFile function reads the input matrix dimensions (Height H and Width W) and matrix values from input.txt file. First and second line in input.txt is H and W.
   - The readFilterFile function reads the filter size R (first line in filter.txt) and filter values from filter.txt file.
2. **Memory Allocation:**
   - Host memory (h_input, h_filter, h_output) is allocated using malloc for storing input, filter, and output matrices.
   - Device memory (d_input, d_filter, d_output) is allocated using cudaMalloc for V1 of the code and cudaMallocManaged for V2.
3. **Data Transfer to GPU:**
   - Input and filter matrices are copied to GPU memory using cudaMemcpy.
4. **Kernel Execution:**
   - The kernel is launched with a grid of blocks, each containing a 16*16 = 256 thread configuration.
   - The kernel computes the convolution operation by calling the kernel function.
5. **Copy Results Back & Cleanup:**
   - The computed output is transferred from the GPU to the host using cudaMemcpy.
   - Output is displayed and using the cmdline stored in output.txt.
   - Memory is freed on both the host using free() and device using cudaFree().
6. **Timing Results**
   - CUDA events are used to measure execution time.
   - CudaEventRecord is used to record the start and stop time for the kernel
   - Memory is freed for these timing events.

CUDA Kernel Implementation

The kernel (conv2d_kernel) does the following:

1. **Thread Identification:** Each thread computes the convolution for one pixel in the output matrix, using blockIdx(The block index in the grid), threadIdx(The thread index within the block), and blockDim(The number of threads per block.).
2. **Zero-Padding Calculation:** The kernel determines the zero-padding size $P = (R-1)/2$ to center the filter.
3. **Performing Convolution**

   Each thread performs the following steps to compute the output value at (x, y):

   - Initialize a sum accumulator to zero.
   - Loop through each element in the $R \times R$ filter.
   - Compute the corresponding input matrix coordinates: $imgX=col+j-P$, $imgY=row+i-P$ where i and j iterate over the filter dimensions.
   - If (imgX, imgY) is within the valid input matrix bounds ($0 \leq imgX < W$ and $0 \leq imgY < H$), multiply the input value by the corresponding filter weight:

     $sum+=input[imgY \times W+imgX] \times filter[i \times R+j]$
   - Store the final sum in the output matrix at (row, col) in 1 D can be written as row* width +column and keep iterating.

TIMING RESULTS for CudaMalloc vs CudaMallocManaged:

Input and Filter used for timing results are **input4** and **filter4.**

Timing Results for **cudaMalloc**(conv2dV1) code for three runs:

GPU kernel took: 58.5155 milliseconds to execute

GPU kernel took: 59.7606 milliseconds to execute

GPU kernel took: 59.8804 milliseconds to execute

Average Time : **59.3855 milliseconds.**

Timing Results for **cudaMallocManaged** (conv2dV2) code for three runs:

GPU kernel took: 40.1787 milliseconds to execute

GPU kernel took: 39.6278 milliseconds to execute

GPU kernel took: 40.7716 milliseconds to execute

Average time: **40.1927 milliseconds.**

Part B: GPGPU-Sim Simulator

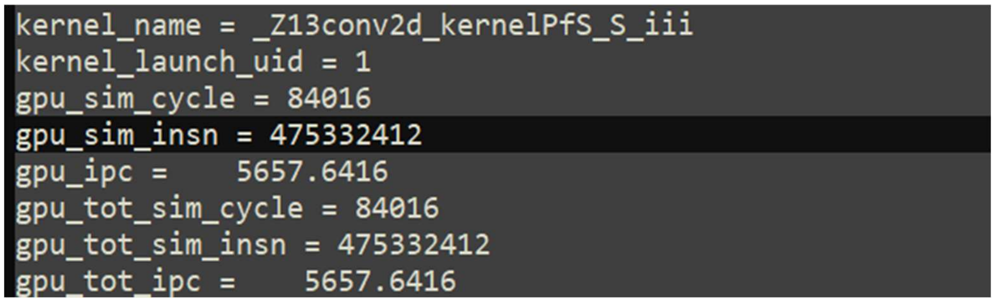
Input used: **input3.txt** and filter used filter3.txt and was verified functionally.

Here are the performance parameters:

a. Total **IPC** of the program = Total number of instructions / Total cycles

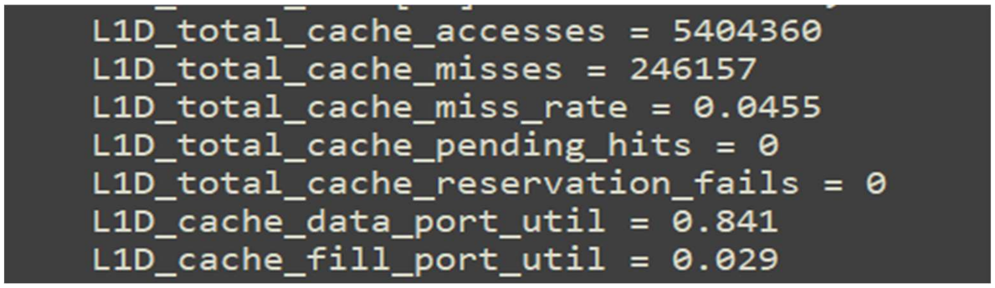
= 475332412/84016

Total IPC = **5657.6416**



```
kernel_name = _Z13conv2d_kernelPfS_S_iii
kernel_launch_uid = 1
gpu_sim_cycle = 84016
gpu_sim_insn = 475332412
gpu_ipc =    5657.6416
gpu_tot_sim_cycle = 84016
gpu_tot_sim_insn = 475332412
gpu_tot_ipc =    5657.6416
```

b. Data cache miss rate = L1D total cache misses/ L1D total cache accesses
= 246157/5404360
= **0.0455**



```
    L1D_total_cache_accesses = 5404360
    L1D_total_cache_misses = 246157
    L1D_total_cache_miss_rate = 0.0455
    L1D_total_cache_pending_hits = 0
    L1D_total_cache_reservation_fails = 0
    L1D_cache_data_port_util = 0.841
    L1D_cache_fill_port_util = 0.029
```