ECE 408- Applied Parallel Programming Project

Hwanseo Choi (hwanseo2),
Simeng Liu (simengl2),
Chandan Vempati (vempati2).

April 17, 2018

Chapter 1

Milestone I

Question 1

TEAM INFO:
Team name : $o - {}_{-} - O_{-} - {}_{-}O - {}_{-} - O$,
Team Members:
Hwanseo Choi (hwanseo2),
Simeng Liu (simengl2),
Chandan Vempati(vempati2),
.

Question 2

The list of kernels and some their corresponding parameters which were measured are shown in Table 1.

Name	Time(%)	Time	Calls
void fermiPlusCgemmLDS128 _batched	34.09	118.48 ms	9
void cudnn::detail::implicit _convolve _sgemm	27.02	93.917 ms	1
void fft2d _c2r _32x32	12.62	43.847 ms	9
sgemm _sm35 _ldg _tn _128x8x256x16x32	8.20	28.512 ms	1
CUDA _memcpy _HtoD	6.46	22.445 ms	14
void cudnn::detail::activation _fw _4d _kernel	4.07	14.158 ms	2
void cudnn::detail::pooling _fw _4d _kernel	3.82	13.292 ms	1
void fft2d _r2c _32x32	1.72	5.9651 ms	9
sgemm _sm35 _ldg _tn _64x16x128x8x32	1.17	4.0583 ms	1
void mshadow::cuda::MapPlanLargeKernel	0.37	1.2844 ms	1
void mshadow::cuda::SoftmaxKernel	0.32	1.1046 ms	1
void mshadow::cuda::MapPlanKernel	0.05	177.02 μs	13
void mshadow::cuda::MapPlanKernel	0.04	146.34 μs	2
sgemm _sm35 _ldg _tn _32x16x64x8x16	0.04	130.11 μs	1
void mshadow::cuda::MapPlanKernel	0.01	22.399 μs	1
void fft2d _r2c _32x32	0.01	20.671 μs	1
CUDA _memcpy _DtoH	0.00	9.9200 μs	1

Table 1.1: CUDA kernel calls and their corresponding parameters

Question 3

The list of APIs and some their corresponding parameters which were measured are shown in Table 1.

Name	Time(%)	Time	Calls
cudaStreamCreateWithFlags	43.55	1.92546 s	18
cudaFree	27.11	1.19848 s	10
cudaMemGetInfo	20.70	915.17 ms	27
cudaStreamSynchronize	7.31	323.39 ms	29
cudaMemcpy2DAsync	1.01	44.605 ms	9
cudaMalloc	0.16	7.2049 ms	45
cudaStreamCreate	0.03	1.5196 ms	4
cuDeviceTotalMem	0.03	1.3522 ms	4
cuDeviceGetAttribute	0.03	1.1863 ms	352
cudaEventCreateWithFlags	0.02	1.0891 ms	114
cudaLaunch	0.02	728.51 μs	53
cudaMemcpy	0.01	405.96 μs	6
cudaSetupArgument	0.01	352.84 μs	619
cudaDeviceGetAttribute	0.00	135.99 μs	116
cuDeviceGetName	0.00	102.82 μs	4
cudaSetDevice	0.00	82.812 μs	35
cudaStreamWaitEvent	0.00	55.338 μs	27
cudaStreamCreateWithPriority	0.00	50.246 μs	2
cudaConfigureCall	0.00	48.487 μs	53
cudaGetDevice	0.00	26.507 μs	10
cudaEventRecord	0.00	21.586 μs	12
cudaGetLastError	0.00	20.909 μs	34
cudaBindTexture	0.00	15.628 μs	1
cudaPeekAtLastError	0.00	12.229 μs	18
cuDeviceGetCount	0.00	6.8830 μs	6
cudaEventCreate	0.00	5.9220 μs	1
cudaStreamGetPriority	0.00	5.8400 μs	1
cuDeviceGet	0.00	5.0390 μs	6
cudaDeviceGetStreamPriorityRange	0.00	4.9430 μs	2
cuInit	0.00	3.8850 μs	3
cuDriverGetVersion	0.00	2.6300 μs	3
cudaEventDestroy	0.00	2.4120 μs	1
cudaGetDeviceCount	0.00	2.1210 μs	1
cudaUnbindTexture	0.00	1.8500 μs	1

Table 1.2: CUDA APIs calls and their corresponding parameters

Question 4

Application Program Interfaces or **APIs** is a set of subroutine definitions that is provided by NVIDIA as part of the CUDA tool and is in-charge of connecting host and device for various purposes. For example, copy memory from host to device or the other direction.

Kernels are the custom code that is defined by the user and are executed N times in parallel by N different CUDA threads, as opposed to only once like regular C functions.

Question 5

Output of rai running MXNET on the CPU is as follows:

```
 Loading fashion-mnist data...
done
Loading model...
done
New Inference
EvalMetric:  'accuracy':  0.8444
```

Question 6

The program run time on the CPU is **12.74 s**.

Question 7

Output of rai running MXNET on the GPU is as follows:

```
 Running /usr/bin/time python m1.2.py
Loading fashion-mnist data...
done
Loading model...
src/operator/././ cudnn _algoreg-inl.h:112:  Running performance tests to
find the best convolution algorithm, this can take a while...(setting env
variable MXNET_CUDNN _AUTOTUNE _DEFAULT to 0 to disable)
done
New Inference
EvalMetric:  'accuracy':  0.8444
```

Question 8

The program run time on the GPU is **2.13 s**.

Chapter 2

Milestone II

Question 1

The list of total execution time for all parameters is shown in Table 3.

Number of Images	Time (s)
10000 (default)	30.58 user 1.55 system 0:30.03 elapsed
1000	1.06 user 0.61 system 0:01.02 elapsed
100	0.70 user 0.48 system 0:00.72 elapsed

Table 2.1: Total execution times and their corresponding parameters

Question 2

The list of op time for all parameters is shown in Table 4.

Number of Images	Op Time 1 (s)	Op Time 2 (s)
10000 (default)	6.61	19.48
1000	0.07	0.20
100	0.01	0.02

Table 2.2: Total execution times and their corresponding parameters

The Op Time scales linearly with the number of images.

Chapter 3

Milestone III

The output of the parallelized code is as follows:

```
 * Running python m3.1.py
Loading fashion-mnist data...
done
Loading model...
done
New Inference
Op Time:  0.293913
Op Time:  0.720379
Correctness:  0.8451 Model:  ece408
* Running python m3.1.py 10
Loading fashion-mnist data...
done
Loading model...
done
New Inference
Op Time:  0.000235
Op Time:  0.000703
Correctness:  1.0 Model:  ece408
* Running python m3.1.py 100
Loading fashion-mnist data...
done
Loading model...
done
New Inference
Op Time:  0.002745
Op Time:  0.008985
Correctness:  0.88 Model:  ece408
```

```
 void mshadow::cuda::MapPlanKernel, float>,
mshadow::expr::Plan, mshadow::Tensor, float, int=1>,
float>>(mshadow::gpu, unsigned int, mshadow::Shape, int=4)
1.25\% 198.01us 14 14.143us 2.0480us 75.071us void
mshadow::cuda::MapPlanKernel, float>, mshadow::expr::Plan,
float>>(mshadow::gpu, unsigned int, mshadow::Shape, int=2)
1.01% 160.13us 2 80.063us 8.2560us 151.87us void
cudnn::detail::activation_fw_4d_kernel>(cudnnTensorStruct, float
const , cudnn::detail::activation_fw_4d_kernel>,
cudnnTensorStruct, float, cudnnTensorStruct*, int, cudnnTensorStruct*)
0.89% 141.66us 1 141.66us 141.66us 141.66us void
cudnn::detail::pooling_fw_4d_kernel, int=0>\\(cudnnTensorStruct, float const
, cudnn::detail::pooling_fw_4d_kernel,
int=0>, cudnnTensorStruct, cudnnPoolingStruct, float, cudnnPoolingStruct,
int, cudnn::reduced_divisor, float)
0.25% 40.255us 1 40.255us 40.255us 40.255us sgemm_sm35_ldg_tn_32x16x64x8x16
0.12% 18.560us 1 18.560us 18.560us 18.560us void mshadow::cuda::SoftmaxKernel,
float>, mshadow::expr::Plan, float>>(mshadow::gpu,
int=2, unsigned int)
0.08% 12.000us 1 12.000us 12.000us 12.000us void
mshadow::cuda::MapPlanKernel, float>,
mshadow::expr::Plan, float, int=3, bool=1, int=2>,
float
(mshadow::gpu, unsigned int, mshadow::Shape, int=2)
0.05% 7.3590us 2 3.6790us 3.0080us 4.3510us void
mshadow::cuda::MapPlanKernel, float>,
mshadow::expr::Plan, float, int=2, int=1>, float>>(mshadow::gpu, unsigned
int, mshadow::Shape, int=2)
0.04% 5.8870us 1 5.8870us 5.8870us 5.8870us [CUDA memcpy DtoH]
0.02% 3.8720us 1 3.8720us 3.8720us 3.8720us void
scal_kernel
(cublasTransposeParams, float const , float, float const *)
0.01% 1.5040us 1 1.5040us 1.5040us 1.5040us [CUDA memset]
```
The subsequent output has been tabulated. We refer you to Table (3.1).

Name	Time (s)	Calls	Avg (ms)	Min (μs)	Max (ms)
cudaStreamCreateWithFlags	1.54822s	16	96.764	19.482	773.80
cudaFree	1.18079s	10	118.08	1.1940	315.38
cudaMemGetInfo	1.00035s	27	37.050	337.28	991.24
cudaDeviceSynchronize	13.589	6	2.2648	7.9570	10.192
cudaMalloc	3.6162	45	80.359	12.882	333.55
cudaMemcpy2DAsync	2.1451	9	238.35	14.825	842.96
cudaStreamSynchronize	1.7335	29	59.775	6.4650	623.37
cuDeviceTotalMem	1.4029	4	350.73	340.77	375.89
cuDeviceGetAttribute	973.49	352	2.7650	516e-3	72.454
cudaEventCreateWithFlags	796.00	112	7.1070	900e-3	308.82
cudaLaunch	580.81	28	20.743	8.5320	55.081
cudaMemcpy	420.03	6	70.005	26.248	151.75
cudaStreamCreate	207.98	4	51.996	24.159	81.876
cuDeviceGetName	119.21	4	29.801	19.015	38.011
cudaSetupArgument	107.69	158	681e-3	523e-3	1.5900
cudaDeviceGetAttribute	99.049	104	952e-3	694e-3	2.3270
cudaSetDevice	91.029	34	2.6770	927e-3	9.8460
cudaMemsetAsync	50.096	1	50.096	50.096	50.096
cudaStreamCreateWithPriority	40.683	2	20.341	20.032	20.651
cudaConfigureCall	39.980	28	1.4270	672e-3	3.5080
cudaGetDevice	33.473	10	3.3470	1.5450	8.5850
cudaPeekAtLastError	16.071	20	803e-3	608e-3	1.0910
cudaGetLastError	8.0830	8	1.0100	557e-3	2.6470
cudaEventQuery	7.4510	1	7.4510	7.4510	7.4510
cuDeviceGetCount	5.9890	6	998e-3	517e-3	2.1960
cuDeviceGet	5.8330	6	972e-3	651e-3	1.2940
cudaDeviceGetStreamPriorityRange	3.9960	2	1.9980	1.5990	2.3970
cuInit	3.8320	3	1.2770	1.1680	1.4220
cuDriverGetVersion	3.1750	3	1.0580	992e-3	1.1300
cudaEventRecord	2.6950	1	2.6950	2.6950	2.6950
cudaGetDeviceCount	2.1030	1	2.1030	2.1030	2.1030

Table 3.1: CUDA APIs calls and their corresponding parameters

The parameters correspoding to the kernel call are as follows:

```
Time(%) Time Calls Avg Min Max Name

92.46% 1.32146s 2 660.73ms 307.68ms 1.01378s mxnet::op::forward_kernel(float*, float const
*, float const *, int, int, int, int, int, int)

⋆ Running nvprof python m3.1.py 10

48.77% 1.3694ms 2 684.69us 319.74us 1.0496ms mxnet::op::forward_kernel(float*, float const
*, float const *, int, int, int, int, int, int)

⋆ Running nvprof python m3.1.py 100

83.74% 13.262ms 2 6.6312ms 3.0892ms 10.173ms mxnet::op::forward_kernel(float*, float
const *, float const *, int, int, int, int, int, int)

6.70% 1.0615ms 14 75.818us 1.5680us 813.94us [CUDA memcpy HtoD]
```

```
⋆ Running nvprof python m3.1.py

Time(%) Time Calls Avg Min Max Name

92.46% 1.32146s 2 660.73ms 307.68ms 1.01378s mxnet::op::forward_kernel(float*,
float const *, float const *, int, int, int, int, int, int)

⋆ Running nvprof python m3.1.py 10

48.77% 1.3694ms 2 684.69us 319.74us 1.0496ms mxnet::op::forward_kernel(float*,
float const *, float const *, int, int, int, int, int, int)

⋆ Running nvprof python m3.1.py 100

83.74% 13.262ms 2 6.6312ms 3.0892ms 10.173ms mxnet::op::forward_kernel(float*,
float const *, float const *, int, int, int, int, int, int)

6.70% 1.0615ms 14 75.818us 1.5680us 813.94us [CUDA memcpy HtoD]
```