Walchand College of Engineering, Sangli
Department of Computer Science and Engineering
Final Year: High Performance Computing Lab 2022-23 Sem I
Class: Final Year (Computer Science and Engineering)
Year: 2022-23
Course: High Performance Computing Lab

Practical No. 10

PRN: **2019BTECS00035**
Title: **CUDA Programming**

---

1. Implement Matrix-matrix Multiplication using global memory in CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

Link:
https://github.com/Dnyaneshwar-dev/HPC-Lab-Sem1/blob/main/Assignment%2010/matrix-mult-global.cu

<<<1,1>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls     Average       Minimum      Maximum           Name
 -------  ---------------  ---------  -------------  -----------  -----------  --------------------
    97.5      17603862851          1  17603862851.0  17603862851  17603862851  cudaDeviceSynchronize
     2.5        456236713          3    152078904.3        17528    456144775  cudaMallocManaged
     0.0           967954          3       322651.3       250554       415898  cudaFree
     0.0           216125          1       216125.0       216125       216125  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances     Average       Minimum      Maximum                          Name
 -------  ---------------  ---------  -------------  -----------  -----------  -----------------------------------
------------
   100.0      17603948659          1  17603948659.0  17603948659  17603948659  matrixMultiply(float*, float*, float
*, int, int)
```

<<<2,4>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average      Minimum     Maximum            Name
 -------  ---------------  ---------  ------------  ----------  ----------  ---------------------
    90.0       2358090715          1  2358090715.0  2358090715  2358090715  cudaDeviceSynchronize
    10.0        262263497          3    87421165.7       17304   262163694  cudaMallocManaged
     0.0           934002          3      311334.0      244295      400852  cudaFree
     0.0            63590          1       63590.0       63590       63590  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average      Minimum     Maximum                          Name
 -------  ---------------  ---------  ------------  ----------  ----------  ----------------------------------------
---------
   100.0       2358080451          1  2358080451.0  2358080451  2358080451  matrixMultiply(float*, float*, float*,
int, int)
```

<<<8,32>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  ---------------------
    59.9        260104982          3   86701660.7      11851  260048905  cudaMallocManaged
    39.8        172888012          1  172888012.0  172888012  172888012  cudaDeviceSynchronize
     0.3          1367358          3     455786.0     362288     527116  cudaFree
     0.0            52676          1      52676.0      52676      52676  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average     Minimum    Maximum                        Name
 -------  ---------------  ---------  -----------  ---------  ---------  ------------------------------------------
------
   100.0        172874260          1  172874260.0  172874260  172874260  matrixMultiply(float*, float*, float*, in
t, int)
```

2. Implement Matrix-Matrix Multiplication using shared memory in CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

Link:

https://github.com/Dnyaneshwar-dev/HPC-Lab-Sem1/blob/main/Assignment%2010/matrix-mult-shared.cu

<<<1,1>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls     Average       Minimum      Maximum           Name
 -------  ---------------  ---------  -------------  -----------  -----------  ---------------------
    98.6      17832282302          1  17832282302.0  17832282302  17832282302  cudaDeviceSynchronize
     1.4        257490518          3     85830172.7        18556    257395847  cudaMallocManaged
     0.0          1006901          3       335633.7       252898       464668  cudaFree
     0.0            66076          1        66076.0        66076        66076  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances     Average       Minimum      Maximum                            Name
 -------  ---------------  ---------  -------------  -----------  -----------  -----------------------------------
------------
   100.0      17832271711          1  17832271711.0  17832271711  17832271711  matrixMultiply(float*, float*, float
*, int, int)
```

<<<2,4>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average      Minimum     Maximum           Name
 -------  ---------------  ---------  ------------  ----------  ----------  ---------------------
    90.2       2336132183          1  2336132183.0  2336132183  2336132183  cudaDeviceSynchronize
     9.7        252101476          3    84033825.3       11986   252049153  cudaMallocManaged
     0.0           948779          3      316259.7      235648      408710  cudaFree
     0.0            45984          1       45984.0       45984       45984  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average      Minimum     Maximum                          Name
 -------  ---------------  ---------  ------------  ----------  ----------  ---------------------------------------
---------
   100.0       2336118396          1  2336118396.0  2336118396  2336118396  matrixMultiply(float*, float*, float*,
int, int)
```

<<<8,32>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    60.8        253054269          3   84351423.0      27415  252907993  cudaMallocManaged
    38.5        160296268          1  160296268.0  160296268  160296268  cudaDeviceSynchronize
     0.7          2857750          3     952583.3     513064    1499833  cudaFree
     0.0           131710          1     131710.0     131710     131710  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average     Minimum    Maximum                          Name
 -------  ---------------  ---------  -----------  ---------  ---------  ----------------------------------------
------
   100.0        160289639          1  160289639.0  160289639  160289639  matrixMultiply(float*, float*, float*, in
t, int)
```

3. Implement Prefix sum using CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

Link:

https://github.com/Dnyaneshwar-dev/HPC-Lab-Sem1/blob/main/Assignment%2010/prefix-sum.cu

<<<1,1>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    56.0        272468280          4   68117070.0       6205  272352977  cudaMallocManaged
    43.8        213044077          1  213044077.0  213044077  213044077  cudaDeviceSynchronize
     0.2           867008          4     216752.0      18471     476186  cudaFree
     0.1           342581          1     342581.0     342581     342581  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average     Minimum    Maximum                        Name
 -------  ---------------  ---------  -----------  ---------  ---------  ----------------------------------------
----
   100.0        213030832          1  213030832.0  213030832  213030832  prefixSum(float*, float*, float*, float*,
int)
```

<<<1,32>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  --------  ---------  ---------------------
    94.1        419810388          4  104952597.0     14652  419567914  cudaMallocManaged
     5.3         23787715          1   23787715.0  23787715   23787715  cudaDeviceSynchronize
     0.4          1738409          4     434602.3     35540     910844  cudaFree
     0.2           708542          1     708542.0    708542     708542  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                        Name
 -------  ---------------  ---------  ----------  --------  --------  ---------------------------------------------
-
   100.0         23780876          1  23780876.0  23780876  23780876  prefixSum(float*, float*, float*, float*, in
t)
```

<<<1,64>>>

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    92.6        259886563          4  64971640.8      5607  259762681  cudaMallocManaged
     6.9         19435545          1  19435545.0  19435545   19435545  cudaDeviceSynchronize
     0.3           963374          4    240843.5     26660     508137  cudaFree
     0.1           365047          1    365047.0    365047     365047  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                        Name
 -------  ---------------  ---------  ----------  --------  --------  ---------------------------------------------
-
   100.0         19427656          1  19427656.0  19427656  19427656  prefixSum(float*, float*, float*, float*, in
t)


CUDA Memory Operation Statistics (by time):
```

4. Implement 2D Convolution using shared memory using CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

Link:

https://github.com/Dnyaneshwar-dev/HPC-Lab-Sem1/blob/main/Assignment%2010/convolution.cu

Threads: 1

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.6        257739899          3  85913299.7    18701  257660598  cudaMallocManaged
     0.3           815191          3    271730.3    42490     456648  cudaFree
     0.0           120758          1    120758.0   120758     120758  cudaMemcpyToSymbol
     0.0            37957          1     37957.0    37957      37957  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum               Name
 -------  ---------------  ---------  ----------  --------  --------  -------------------------------
   100.0         16304415          1  16304415.0  16304415  16304415  convolution_2d(int*, int*, int)


CUDA Memory Operation Statistics (by time):
```

Threads: 32

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.6        257111316          3  85703772.0    18275  257031754  cudaMallocManaged
     0.3           825387          3    275129.0    41599     483129  cudaFree
     0.1           133246          1    133246.0   133246     133246  cudaMemcpyToSymbol
     0.0            29602          1     29602.0    29602      29602  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum               Name
 -------  ---------------  ---------  ---------  -------  -------  -------------------------------
   100.0          4014970          1  4014970.0  4014970  4014970  convolution_2d(int*, int*, int)


CUDA Memory Operation Statistics (by time):
```

Threads: 256

```
Exported successfully to
/tmp/nsys-report-5002-6fe4-01b0-fe77.sqlite


CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.8        263444022          3  87814674.0    17929  263374459  cudaMallocManaged
     0.2           522316          3    174105.3    38338     315961  cudaFree
     0.0           123902          1    123902.0   123902     123902  cudaMemcpyToSymbol
     0.0             1289          1      1289.0     1289       1289  cudaLaunchKernel


CUDA Memory Operation Statistics (by time):

 Time(%)  Total Time (ns)  Operations  Average  Minimum  Maximum      Operation
 -------  ---------------  ----------  -------  -------  -------  ------------------
   100.0             2432           1   2432.0     2432     2432  [CUDA memcpy DtoD]
```