Name: Brijesh Mavani
CWID: A20406960
University: Illinois Institute of Technology
Course: Parallel and Distributed Processing
Assignment: 5 - Result Report

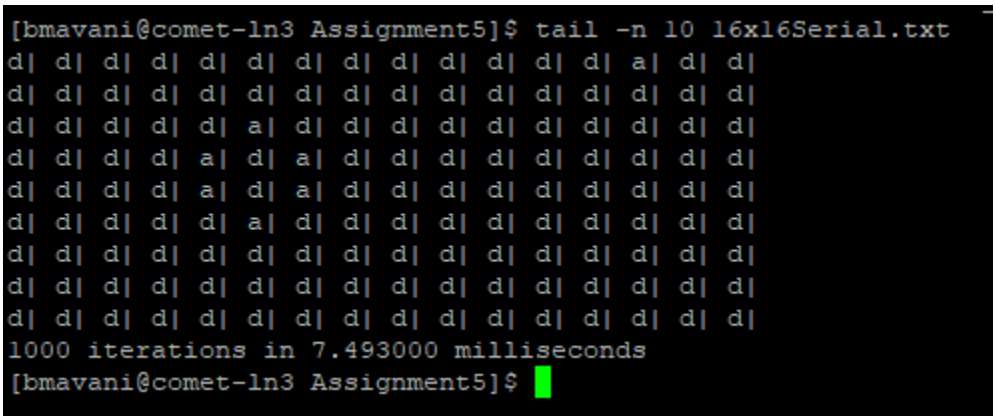
The results of the execution of the code are below. The speed up (Sp) can be calculated by Ts/Tp.

Efficiency (Ep) = Sp/P where P is the number of processors. Here, we will take GPU node count as the number of processors for efficiency calculations.

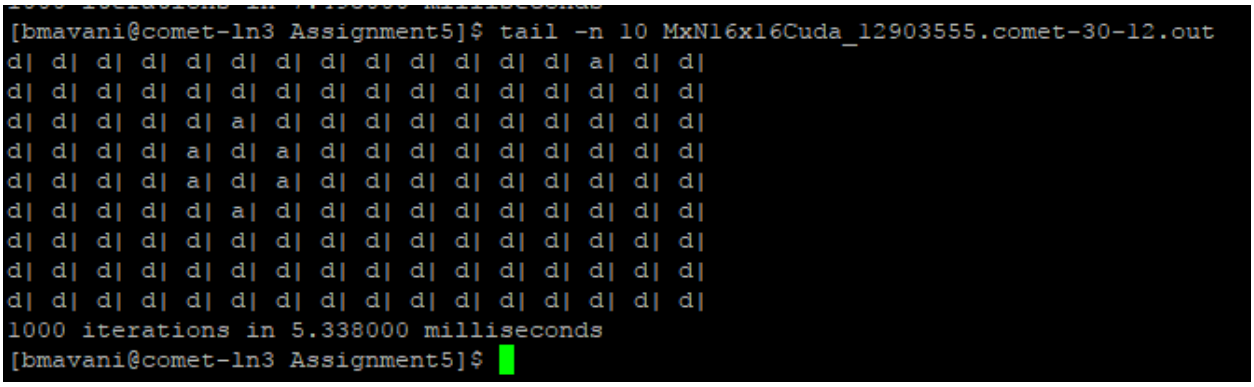
Note: Resulting log files are attached in the main zip folder. As log file size is huge (585.9MB) for matrix size 10000x10000 for serial as well as parallel exeution, those files are not uploaded to blackboard due to size limit. Those are present on comet with all other files at path: /home/bmavani/Assignment5

- 16x16 Matrix with 1000 iterations:

1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 7.493000 milliseconds.



2) Parallel implementation: Tp = 5.338000 milliseconds



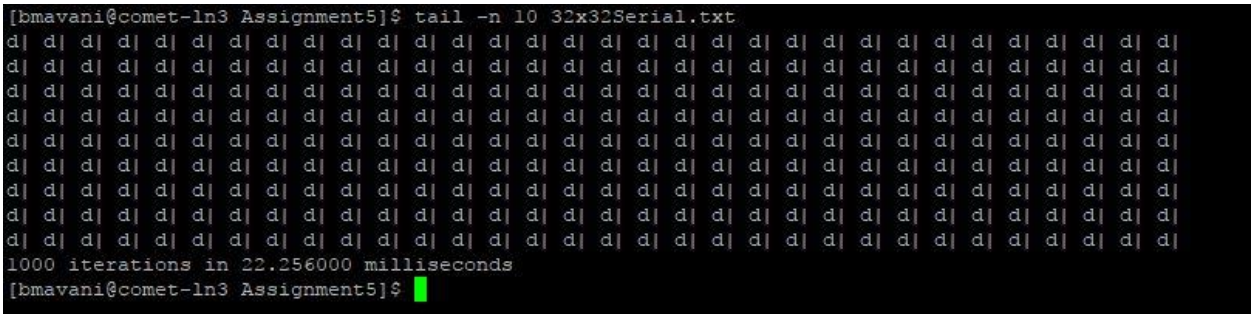
Sp = 7.493000/5.338000 = 1.404. Note that the reported speedup of the GPU is over the serial CPU implementation.

I have used “#SBATCH --gres=gpu:k80:4” command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

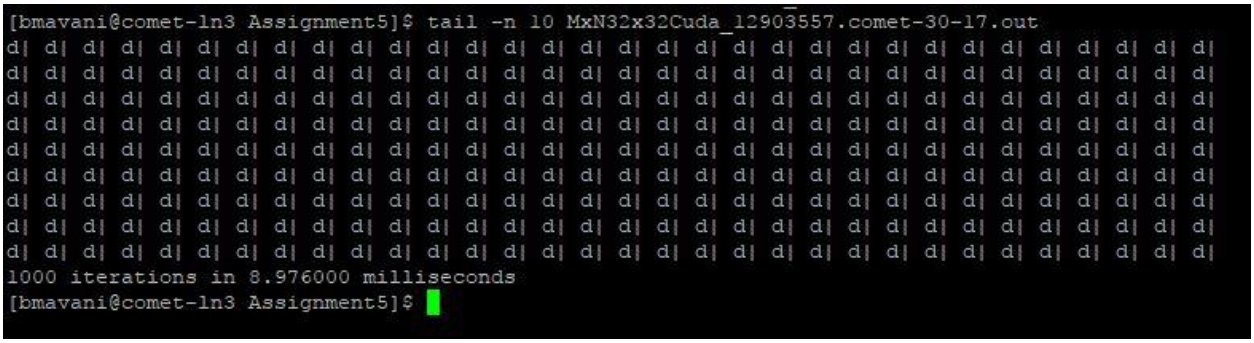
Ep = 1.404 / 4 = 0.35

- 32x32 Matrix with 1000 iterations:

1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 22.256000 milliseconds.



2) Parallel implementation: As this is executed in parallel, we can consider it as a Tp. Tp = 8.976000 milliseconds



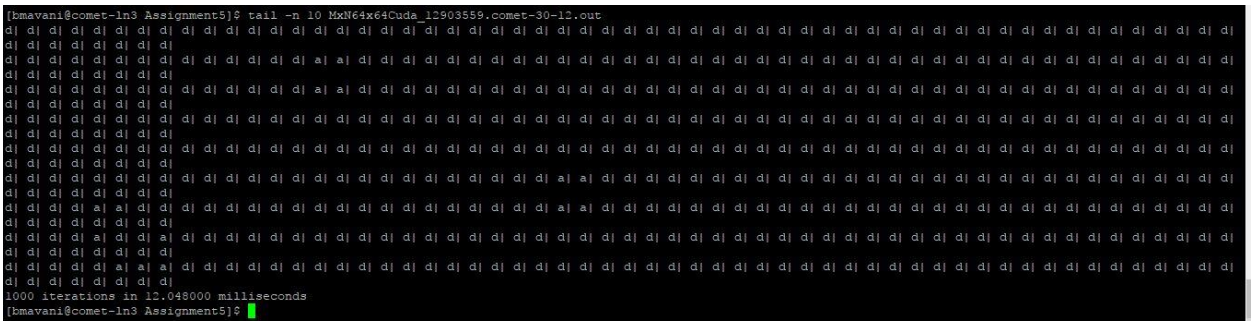
Sp = 22.256000/8.976000 = 2.480. Note that the reported speedup of the GPU is over the serial CPU implementation.

I have used “#SBATCH --gres=gpu:k80:4” command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

Ep = 2.480 /4 = 0.62

- 64x64 Matrix with 1000 iterations:
  1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 91.378000 milliseconds.

  2) Parallel implementation: As this is executed in parallel, we can consider it as a Tp. Tp = 12.048000 milliseconds

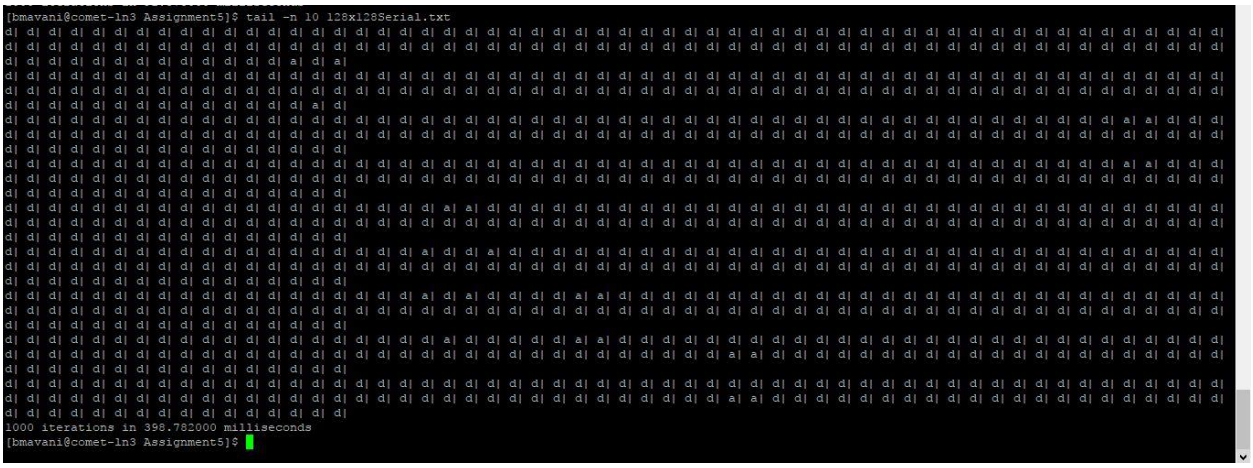


Sp = 91.378000/12.048000 = 7.584. Note that the reported speedup of the GPU is over the serial CPU implementation.

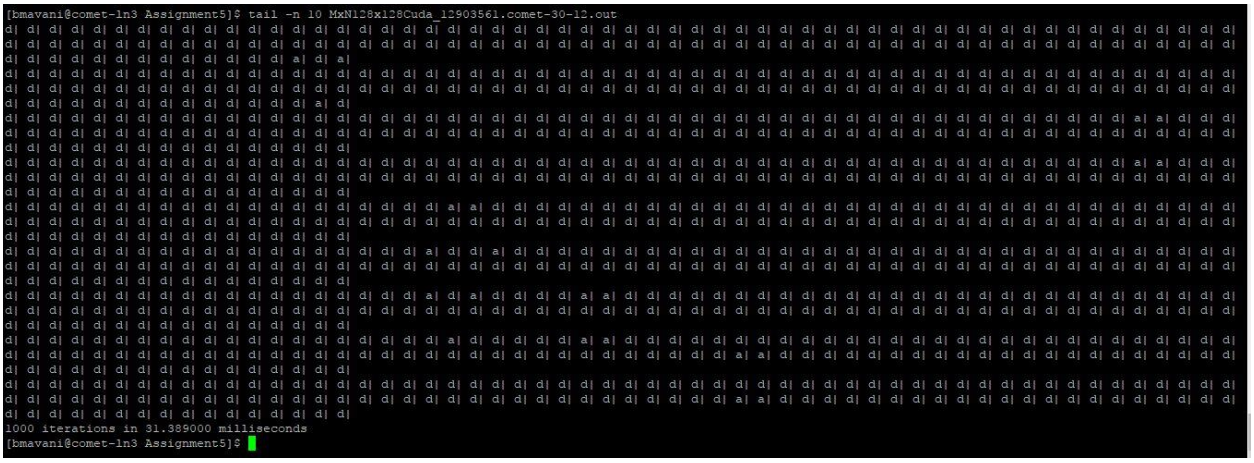
I have used “#SBATCH --gres=gpu:k80:4” command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

Ep = 7.584/ 4 = 1.90

- 128x128 Matrix with 1000 iterations:
  1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 398.782000 milliseconds.



  2) Parallel implementation: As this is executed in parallel, we can consider it as a Tp. Tp = 31.389000 milliseconds

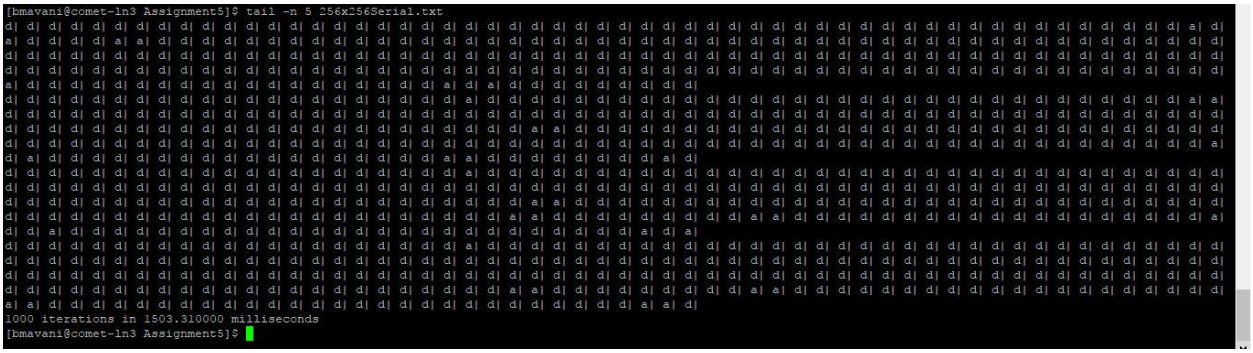


Sp = 398.782000/31.389000 = 12.705. Note that the reported speedup of the GPU is over the serial CPU implementation.

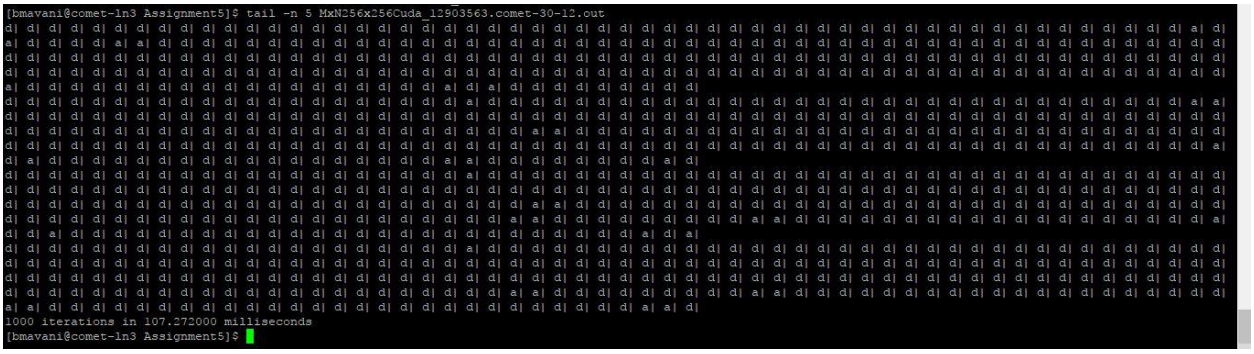
I have used “#SBATCH --gres=gpu:k80:4” command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

Ep = 12.705/ 4 = 3.18

- 256x256 Matrix with 1000 iterations:
  1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 1053.310000 milliseconds.



  2) Parallel implementation: As this is executed in parallel, we can consider it as a Tp. Tp = 107.272000 milliseconds

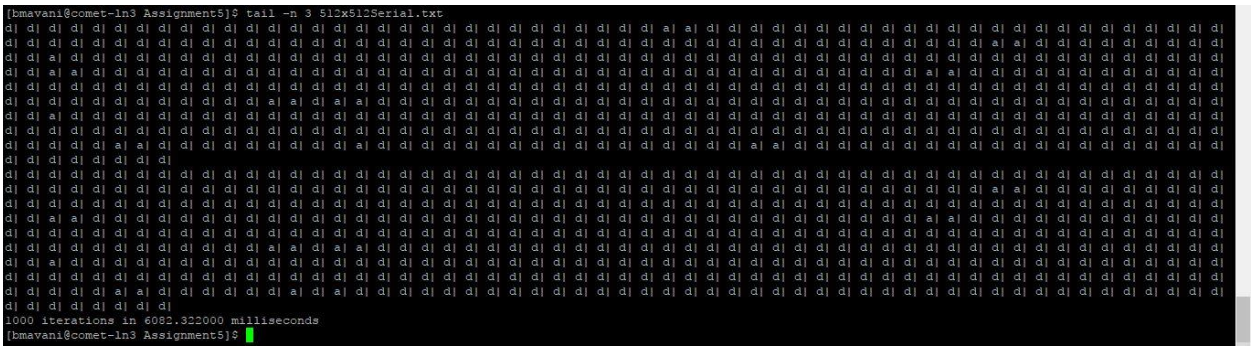


Sp = 1053.310000/107.272000 = 9.819. Note that the reported speedup of the GPU is over the serial CPU implementation.

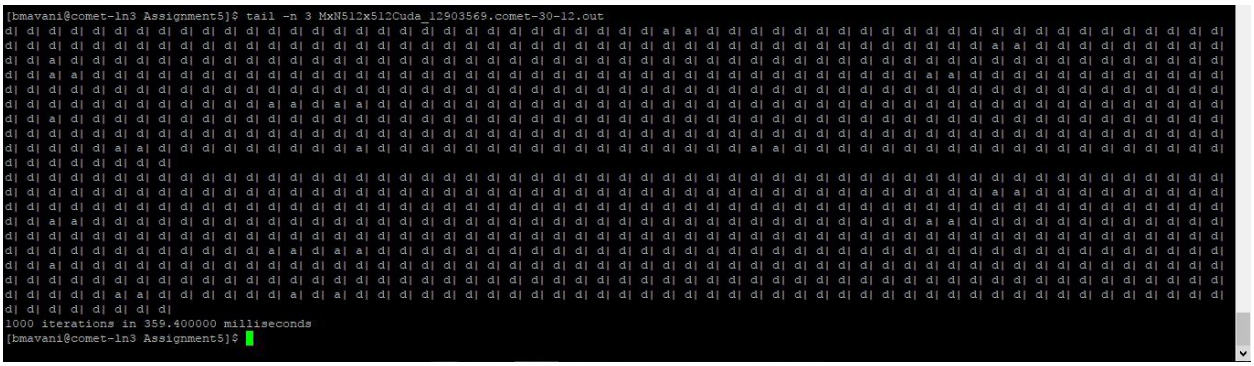
I have used “#SBATCH --gres=gpu:k80:4” command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

Ep = 9.819/4 = 2.45

- 512x512 Matrix with 1000 iterations:
  1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 6082.322000 milliseconds.



  2) Parallel implementation: As this is executed in parallel, we can consider it as a Tp. Tp = 359.400000 milliseconds



Sp = 6082.322000 /359.400000 = 16.924. Note that the reported speedup of the GPU is over the serial CPU implementation.

I have used “#SBATCH --gres=gpu:k80:4” command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

Ep = 16.924/4 = 4.23

- 1024x1024 Matrix with 1000 iterations:
  1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 24124.496000 milliseconds.

  2) Parallel implementation: As this is executed in parallel, we can consider it as a Tp. Tp = 1057.543000 milliseconds

Sp = 24124.496000/1057.543000= 22.812. Note that the reported speedup of the GPU is over the serial CPU implementation.

I have used "#SBATCH --gres=gpu:k80:4" command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

Ep = 22.812/4 = 5.70.

- 10000x10000 Matrix with 1000 iterations:
  1) Serial implementation: As this is executed on a single processor sequentially, we can consider this as Ts. So, Ts= 2341883.766000 milliseconds.

```
[bmavani@comet-ln2 Assignment5]$ tail -n 1 10000x10000Serial.txt
1000 iterations in 2341883.766000 milliseconds
[bmavani@comet-ln2 Assignment5]$
```

  2) Parallel implementation: As this is executed in parallel, we can consider it as a Tp. Tp = 94484.554000 milliseconds

```
[bmavani@comet-ln2 Assignment5]$ tail -n 1 MxN10000x10000_Cuda12914311.comet-30-
09.out
1000 iterations in 94484.554000 milliseconds
[bmavani@comet-ln2 Assignment5]$
```

Sp = 2341883.766000/94484.554000 = 24.786. Note that the reported speedup of the GPU is over the serial CPU implementation.

I have used "#SBATCH --gres=gpu:k80:4" command in sh file which will obtain 4 K80 GPUs. Hence, the code will be executed in parallel on 4 GPUs. So, for efficiency calculations number of processors has been taken as 4.

Ep = 24.786 /4 = 6.20.

Analysis: As we can see in the resulting timings for serial and parallel codes, the GPU is significantly faster than the CPU. The GPU has very fast memory chips with much higher memory throughput compared to the CPU's RAM. The evaluation of the matrix cells is very heavy on memory I/O and not so much on arithmetic operations. Due to this superiority of the GPU over CPU plays a critical part in giving speed up.

- Efficiency
  - ✓ 1024x1024 Matrix using 4 GPU nodes and 1000 iterations: 5.70
  - ✓ 10000x10000 Matrix using 4 GPU nodes and 1000 iterations: 6.20

- Speed up
  - ✓ 1024x1024 Matrix with 1000 iterations: 22.812.
  - ✓ 10000x10000 Matrix with 1000 iterations: 24.786.