Lab 10

CSEN 145L: Parallel Computing Libraries

OpenBLAS is an open-source implementation of Basic Linear Algebra Subprograms (BLAS) optimized for high performance on various CPU architectures, widely used for large-scale linear algebra computations. cuBLAS is NVIDIA's GPU-accelerated counterpart to BLAS, included in the CUDA Toolkit, and provides significant performance enhancements by utilizing the parallel processing capabilities of NVIDIA GPUs. Both libraries are essential for efficiently handling complex mathematical operations in scientific computing, enabling faster computations in fields such as machine learning, numerical simulation, and data analysis.

You are provided with openblas_matmul.cpp, openblas_dotp.cpp, cublas_matmul.cu and cublas_scaling.cu and you need to figure out and call the library API function to carry out necessary computations. A serial code is already provided in the implementation for comparisons.

To get Started, Clone or download the lab10 folder from github.

1. All the implementations are initialized with random values with matrix or vectors.

2. Complete the remaining code

3. To compile/execute and test your code, you will need to request resources from WAVE HPC by either requesting interactive (srun method) or non-interactive (sbatch) resources.

I. OpenBLAS

A. Srun method:

srun --partition=cpu --cpus-per-task=8 --mem=2GB --nodes=1 --ntasks-per-node=1 --pty /bin/bash

After you are allotted the requested resources, your current node will change from login1/2/3 to cpuXX or memXX.

1. You will need to manually install OpenBlas Library using anaconda as it not currently installed on HPC. Follow the commands given below in your home directory.

- ***module load Anaconda3*** (load anaconda to setup env and install packages)
- ***conda create —name OpenBlas*** (create environment)
- ***conda activate OpenBlas*** (your env will change from base to OpenBlas)

```
) (openBlas) [rkachhadiya@wave8-login1 lab10]$
```

- ***conda install -c conda-forge openblas*** (To install openblas)

2. Now to compile and execute your openblas code after completing, you will need to compile it using g++ and link the necessary library paths

To Compile ***openblas_dotp .cpp*** file use,

- **g++ -o openblas_dotp openblas_dotp.cpp -L/WAVE/users2/unix/your_username/.conda/envs/openBlas/lib -I/WAVE/users2/unix/your_username/.conda/envs/openBlas/include -lopenblas**

Export path

- **export LD_LIBRARY_PATH=/WAVE/users2/unix/your_username/.conda/envs/openBlas/lib:$LD_LIBRARY_PATH**

To run your code,

- **./openblas_dotp**

Please follow the above instructions in their order and use the same approach for compiling and running openblas_matmul.cpp program.
Note: Make sure to replace filenames and executable names for other program.

B. Sbatch Method:

Use the openblas_imp.sh file provided to submit your batch job and make sure you change username in your path in the export command and while linking to ensure correct path and then run your program.

II. cuBLAS

A. Srun method:

1. srun --partition=hub --gres gpu:00 --cpus-per-task=8 --mem=8G --nodes=1 --ntasks=1 --time=00:01:00 --pty /bin/bash

2. Module load CUDA

3. To compile your code,

- ***nvcc -o cublas_matmul cublas_matmul.cu -lcublas***
  and
- ***nvcc -o cublas_scale cublas_scale.cu -lcublas***

4. Run your executable

- ***./cublas_matmul***
- ***./cublas_scale***

B. Sbatch Method:

Cublas_imp.sh is provided in the github repo for your reference, make sure you change the filenames before submitting batch job for the both the programs.

Final Deliverables: Submit all 4 files with implementation: openblas_matmul.cpp, openblas_dotp.cpp, cublas_matmul.cu and cublas_scale.cu with their respective terminal output as log file

References

https://docs.nvidia.com/cuda/cublas/
http://www.openmathlib.org/OpenBLAS/