Parallel Programming with MPI

Tutorial Session

MPI

- **Message Passing Interface (MPI)** is a specification for software developers used to make use of a cluster of computers.
- The problem with a computing cluster is that while some of the CPUs share memory (Shared Memory), others have a distributed memory architecture which is only connected by network.
- Today's developers are able to make use of these distributed memory, shared memory and a hybrid system of both; all with the power of MPI.

Pros

- It runs on a distributed memory architecture
- It can be used on a wider range of problems since it exploits both task parallelism and data parallelism
- Each process has its own local variables
- Highly portable with specific optimization for the implementation on most hardware
- Distributed memory computers are less expensive than large shared memory computers

Cons

- It requires more programming changes to go from serial to parallel version
- It can be harder to debug
- Performance is limited by the communication network between the nodes

MPI in Data Science

- Taking advantage of functionality commonly used in high performance computing, a data-parallel environment can be created.

- For example, a distributed Python NumPy library can be developed, which leverage MPI for inter-process communication and data transfers or a distributed implementation of K-Means clustering, showing sensitivity to local process problem sizes and operations that required large amounts of collective communication.

Outlook

- Sometime problems that a firm witnesses are complex and computationally expensive
- Data scientists need to implement parallel programming to solve them or to get the results very quickly.
- It is important to have sound knowledge in parallel programming for a competitive edge in the data science landscape.

Outline of Assignment (MPI)

Assignment (MPI) will cover following topics:

- Apply the standard message-passing algorithms in MPI.
- Study the concept of point-to-point communication, blocking communication, non-blocking communication, and collective communication.

References

Python MPI Introduction

https://nyu-cds.github.io/python-mpi/01-introduction/

MPI Applications

https://www.aspsys.com/solutions/hpc-applications/mpi-applications/

MPI for Python

https://mpi4py.readthedocs.io/en/stable/