Implementation of the parallel I/O operations for big-data applications using message passing interface (MPI)

Abstract

The goal of this research is to implement the Parallel input and output (I/O) operations in big-data applications using message passing interface (MPI) with specific focus on the most common structured file formats like CSV and Apache Parquet. The proposed MPI library functions must be able to deal with large scale and high dimensional datasets. Besides, they must provide interface to implement the MPI IO operations regardless of the type of the input and output file.

Motivation

To ensure efficiency and scalability, the parallel input/output system must provide a high-level interface supporting partitioning of file data among processes and a collective interface supporting complete transfers of global data structures between process memories and files. MPI-2 standard provides a custom interface to support the parallel input/output operations called MPI IO. Currently, MPI IO does not has the ability to perform the parallel IO operations on the more structured file types specifically the file formats from the big data applications like CSV or parquet formats.

Novelty

MPI is a message-passing application programmer interface, which provides the high performance, scalability, and portability. MPI remains the dominant model used in high-performance computing today. MPI for Python provides MPI bindings for the Python language, allowing programmers to exploit multiple processor computing systems. mpi4py is constructed on top of the MPI specifications and provides an object-oriented interface which closely follows MPI-2 C++ bindings. MPI programming by itself doesn’t know anything about the structure of files in the IO operations and views the file as a linear byte stream. On the other hand, the Parquet file format is a common binary data store, used particularly in the Hadoop/big-data applications. Parquet format is an ideal storage mechanism for Python-based big data workflows. Although many of the common cases of parallelization in the big data applications can be solved with MPI4py alone, there are cases where special formatted file structures cannot be read and write within MPI4py infrastructure. MPI4py lib/MPI (C++) is missing the functions for read/write a CSV or parquet file directly. In my research I must study the read/write implementation of parquet file from the spark source code and implement its parallel MPI IO version in python or C++. Although there are python libraries designed for utilizing the parquet file format, this format is designed to work well with systems designed for parallel execution. Studies shows that an MPI solution for a big data problem outperforms the solution by the Apache spark.

Columnar file formats allow more efficient slicing of data (both horizontal and vertical) in the big data applications Therefore, they provide faster IO operations on the parallel big data applications. The most common formats are CSV, JSON, AVRO, Protocol Buffers, Parquet. On the other hand, MPI is a message-passing application programmer interface, which provides the high performance, scalability, and portability. MPI remains the dominant model used in high-performance computing today.

Result

The novel tool which provides the MPI4py based functions for read/write a CSV and parquet file will enable the researchers to apply the MPI solutions directly on the big data problems through the direct parallel IO operation on the big datasets. The proposed IO implementations may show superiority on one dataset but not on another big data applications. On the other hand, regarding the Parquet format’ write operation there is still some significant limitations.