Paper Review

“ MPI-Vector-IO: Parallel I/O and Partitioning for Geospatial Vector Data ”

1. Summary

MPI-Vector-IO is used for large-size, complex, and heterogeneous geospatial datasets. MPI-Vector-IO is a parallel I/O library that uses MPI-IO specifically for partitioning and reading irregular vector data formats such as Well-Known Text. In brief, it uses MPI to optimize parallel Geographic Information systems (GIS).

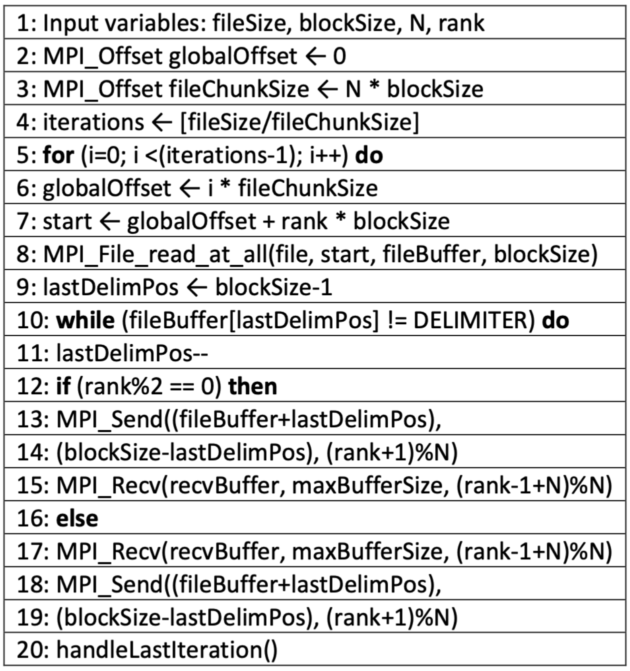
Usually, geographic datasets are divided into grids, which means they can be optimized using partition read and partition write. This paper focuses on the dataset which contains vector data where shapes are represented with points, lines, and polygons. For the data-and compute-intensive work, this paper designs an MPI function that can partition the file and spatial so that it can be processed with MPI processes with efficiency.

The challenges are as follows.

1. How to partition file that contains irregular and unstructured data.
2. Design an MPI-IO function that is effective for geospatial applications.
3. MPI process can not read/write more than 2GB of data in a single operation.

The algorithms are follows.

Algorithm 1 Iterative File Reading - Message based



```
Input variables: fileSize, blockSize, N, rank
MPI_Offset globalOffset ← 0
MPI_Offset fileChunkSize ← N * blockSize
iterations ← [fileSize/fileChunkSize]
for (i=0; i <(iterations-1); i++) do
    globalOffset ← i * fileChunkSize
    start ← globalOffset + rank * blockSize
    MPI_File_read_at_all(file, start, fileBuffer, blockSize)
    lastDelimPos ← blockSize-1
    while (fileBuffer[lastDelimPos] != DELIMITER) do
        lastDelimPos--
    if (rank%2 == 0) then
        MPI_Send((fileBuffer+lastDelimPos),
        (blockSize-lastDelimPos), (rank+1)%N)
        MPI_Recv(recvBuffer, maxBufferSize, (rank-1+N)%N)
    else
        MPI_Recv(recvBuffer, maxBufferSize, (rank-1+N)%N)
        MPI_Send((fileBuffer+lastDelimPos),
        (blockSize-lastDelimPos), (rank+1)%N)
handleLastIteration()
```

For large files, a block size is 2GB thanks to a 32-bit address. Otherwise, the file is logically divided equally among the processes. A subset of cells that correspond to the divided data (which use the halo region to do splitting) is then assigned to the processes to do computing of one grid cell and its contiguous cells. The geometries in a file partition (P_0 to $P_1 1$) are mapped to corresponding grid cells. An all-to-all personalized data exchange produces global spatial partitioning.

It scales well with MPI processes and file size and achieves bandwidth up to 22 GB/s for common spatial data access patterns.

2. Advantages

+ Compared to NetCDF and similar formats, MPI-Vector-IO saves the time of pre-processing and gets real-time performance.
+ Some research uses the MapReduce paradigm such as Hadoop. MPI-Vector-IO can be faster than Hadoop since it can use High-Performance Computers. The MPI-based system is faster than the Hadoop-based system.
+ It is designed to use any parallel filesystem. It is not slow, cumbersome, and does not overwhelm the memory capacity of individual nodes for larger data.

3. Disadvantages

– Lack GPU support.