A MapReduce Based Parallel Processing Framework for Spatial Data [SDPPF]

Nayana Durgada Veerappa
Computer Science Department
San Jose State University
San Jose, CA 95192
408-775-9545
nayana.41091@gmail.com

ABSTRACT

Large volume of geographic data have been, and continue to be, collected with the help of modern data techniques such as global positioning systems (GPS), high-resolution remote sensing, location-aware services and surveys, and internet-based volunteered geographic information. Spatial data processing requires huge amount of calculation and to speed up the process, parallel processing is very much essential. But, for parallelization, popular parallel frameworks need lots of code development, which are difficult for geo-scientists who are often beginners in programming.
In this paper, an easy way of parallel processing framework is proposed and named SDPPF – Spatial Data Parallel Processing Framework. This paper highlights about SDPPF which is a MapReduce based framework and can directly reuse existing binary executable program for parallel processing. An implementation details, architectural details and importance of parallel model of SDPPF are presented and its evaluation is analyzed by testing specific algorithms. Experimental results clearly state that SDPPF is a flexible, easy-to-use and scalable framework for spatial data parallel processing.

1. INTRODUCTION

Spatial Data, Also known as geospatial data or geographic information it is the data or information that identifies the geographic location of features and boundaries on Earth, such as natural or constructed features, oceans, and more. Spatial data is usually stored as coordinates and topology, and is data that can be mapped. [2]

Spatial data processing is very complex and involves analysis large amount of spatial data and non-spatial data, utilizing technologies like GIS, global positioning systems (GPS), high-resolution remote sensing, location-aware services and surveys, and internet-based volunteered geographic information, without the production of a map. It utilizes large volume of geodata to identify problems and provide a reasonable solution to the problems. Spatial data processing involves extracting features, querying the attribute, classification and detection, where as all the steps requires huge amount of calculation.

With the improving technologies like sensor and storage, more and more data can be stored and shared. As and when the dataset is growing, it has outpaced the power of a single processor, thus there is a need for parallel processors. Parallel processing framework has been successful for the past decade, but designing, writing, debugging, parallel programs are difficult for the geo-scientists who are often beginners in programming.

MapReduce programming model reduces the complexity of parallel programming. It allows programmers without parallel programming experience to easily utilize cluster or grid for data-intensive computing. One of an Open Source implementation of MapReduce framework is Hadoop, which is popular in both academic and industrial area. Hadoop also requires lots of code development and users must know the source or implementation of target algorithm. [3]

This paper proposes an easy-to-use parallel processing framework, named spatial data parallel processing framework (SDPPF). SDPPF uses directly the existing executable binary code to do parallel processing with less or no code modification. It also takes care of data partitioning, failure handling, scheduling the task and also communication. It eases the work load of geo-scientists in parallel computing on cluster or grid easily. [3]

2. PARALLEL MODEL

SDPPF makes use of popular and easy-to-use parallel processing model named MapReduce. MapReduce is a distributed parallel programming model that process massive amounts of unstructured data in parallel across a distributed cluster of processors or stand-alone computers. It is first introduced by Google. This model has two steps: Map and Reduce.

Map ( ) processes input data and generates set of key-value pairs. Basically map performs filtering and sorting functions. Reduce ( ) merge all intermediate values with the same intermediate key. Basically it performs summarizing function. MapReduce is implemented on cluster or a grid. It helps in achieving scalability and fault-tolerance for variety of applications. As shown in Fig.1 and Fig. 2

There are two types of nodes: Master and Slave. MapReduce includes three phases in SDPPF: Prepare, Map, Reduce. Master node isolates task workload on slave node. It also takes care of centralized control where as slave nodes concentrate on task completion. [5]

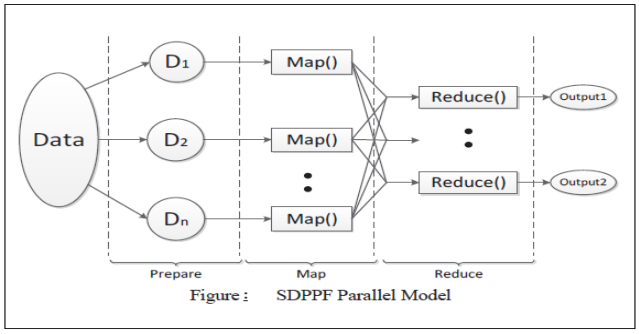

Fig: 1

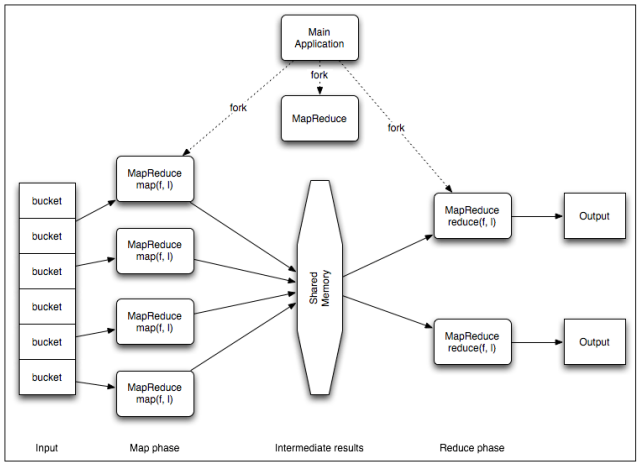

Fig: 2

The three phases prepare, map and reduce corresponds with data partitioning, algorithm processing and result merging. Prepare and reduce phase are processed on master node and map phase is on slave node. In SDPPF, users need to provide three executable programs with functions as prepare, map and reduce only. It can be easily implemented by programmer and the original sequential program can be reused directly or with few modifications in the map phase. [3]

3. ARCHITECTURE & DESIGN

SDPPF is developed and running on SIGRE [6] (Spatial Information Grid Runtime Environment), which is an autonomic runtime environment for geo-computation. SIGRE provides a basic job framework [7] that makes it possible to create, run, monitor, and manage an SIG job through web service interface. With the toolkits provided by SIGRE, it is very simple and easy to develop and deploy an SIG job. [3]

SDPPF is built up by the following components: SIGRE platform; Controller including Preparer, Mapper and Reducer; Dispatcher including Task Scheduler, Fault Handler and Data Manipulator, Processor including Uploader & Downloader, RPC Handler and Exception Handler; Programs for geo-computation. It is shown in Fig. 3. [3]

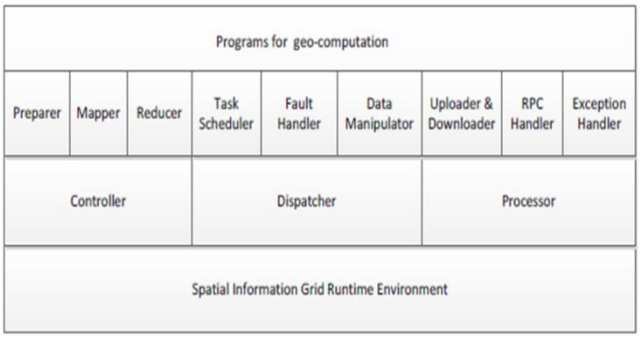

Fig: 3. SDPPF Architecture

SDPPF architecture includes controller, dispatcher and processor. Controller has three sub divisions, Prepare, map and reduce. These are processed in order. Prepare performs functions like retrieving input data from end user via SIGRE interface. It also splits the data and generates instruction files for mapper and reducer use. Map performs functions and produces intermediate key value pairs. These intermediate results are merged by reducer to produce a final output file. [3]

Dispatcher has task scheduler, fault handler and data manipulator as its sub-divisions. Task scheduler manages task scheduling and distribution. It supports strategies like FIFO (first in first out). Before assigning any task to any node, it first checks for available nodes which can run specific task. Data manipulator maintains three queues to store data, Input, Output and Failed. Fault Handler will check failed data queue and re-process the task to be successful. [3]

Processor has Uploader & Downloader, RPC handler and Exception Handler as its part. It runs series of threads on master node which has been assigned by the dispatcher. Uploader set up the FTP connection between remote nodes and sends data files from master to slave node. Once the SIG job is completed, downloader loads intermediate data into master node. RPC handler will add result to the output data queue. Upon any error, exception handler will take care of that record. [3]

4. EVALUATION

In order to evaluate SDPPF, the total execution time of different spatial algorithm is tested in high performance computing cluster with different data volume and computing node count.

First, maximum likelihood classification algorithm (shortly for MLC) is selected for the test. In statistics, **maximum-likelihood estimation** (**MLE**) is a method of estimating the parameters of a statistical model. When applied to a data set and given a statistical model, maximum-likelihood estimation provides estimates for the model's parameters. [8] For example, if I have a data set of penguin features and I want to calculate the height of the penguins. Then I cannot find the height of each penguin thus I will calculate for few and will perform mean and variance to it. So that I can assume that the average height of the penguin would be so and so.

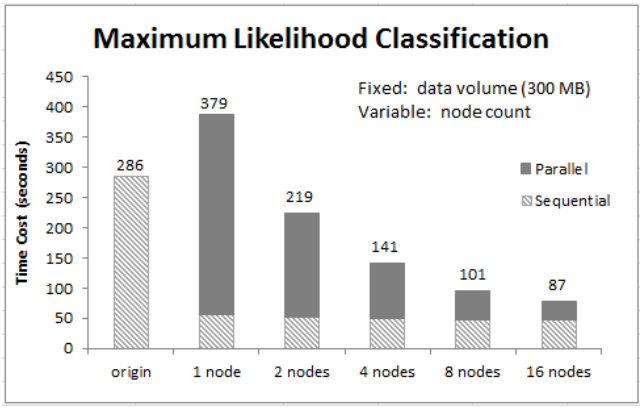

Fig 4: Execution Time vs. Node Count

In Fig 4, Origin represents the execution time for sequential program. Rest of the columns indicate that as the number of nodes increases time cost decreases which clearly states that, for huge volume of data parallel processing will reduce the time cost as and when we increase the number of nodes processing the data.

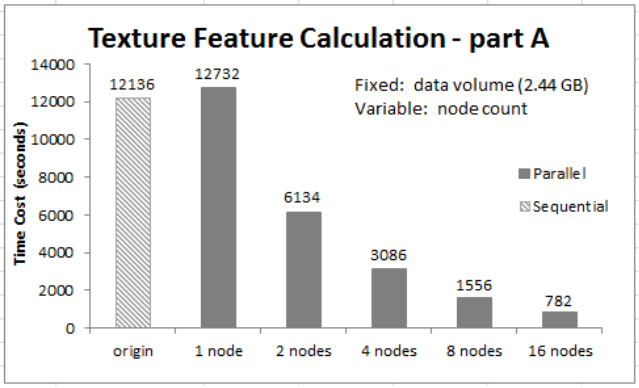

Fig 5: Execution Time vs. Node Count

Second, texture feature calculation algorithm (shortly for TFC) is tested. This algorithm is to find the texture feature of the data. In Fig 5, indicates that the execution time is reduced to half when the number of nodes are doubled. This increases the performance.

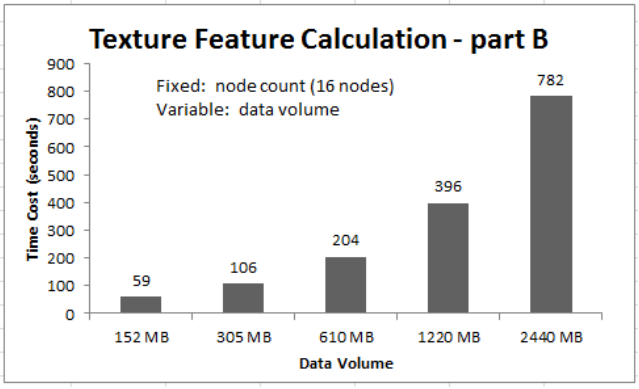

Fig 6: Execution Time vs. Data Volume

In fig 6, Node count is kept constant to 16 and the data volume varies. We can observe that with the increasing volume of data time is also increasing. This indicates the scalability factor of SDPPF.

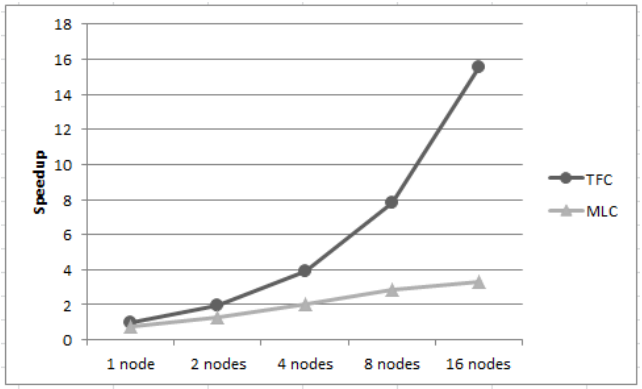

Fig 7: Execution Time vs. Data Volume

Fig. 7 concludes that SDPPF supports parallel processing. With the increasing number of nodes, execution time decreases and speed increases with the help of parallel processing.

5. CONCLUSION

This paper provides an easy-to-use parallel processing framework named SDPPF. It is a MapReduce based parallel processing framework which is implemented on SIGRE. Since Geo-scientists are often beginners in programming, SDPPF can reuse existing binary executable program which requires less or no code modification. This framework eases the workload of Geo-scientists.

SDPPF provides the framework that can support programs running parallel easily and effectively accelerates the processing speed of spatial data with high scalability and low overhead.

REFERENCES

[1] http://en.wikipedia.org/wiki/Spatial_analysis#Spatial_data_analysis.

[2] http://www.webopedia.com/TERM/S/spatial_data.html

[3] Dong Zhao, Zhen Chun Huang (2011) "A MapReduce Based Parallel Processing Framework for Spatial Data-SDPPF" International Conference on Electrical and Control Engineering (ICECE). pp.1258-1261

[4] Diansheng Guo, Jeremy Mennis (2009) "Spatial Data mining and geographical knowledge discovery- An Introduction" Computers, Environment and Urban Systems. Vol.33,pp. 403-408

[5] http://en.wikipedia.org/wiki/MapReduce

[6] ZhenChun Huang, GuoQing Li, Bin Du, Yi Zeng and Lei Gu (2007) SIGRE –An Autonomic Spatial Information Grid Runtime Environment for Geo-computation, ADVANCED PARALLEL PROCESSING TECHNOLOGIES, Volume 48

[7] Huang, Z.C., Li, G.: (2006) An SOA based On-Demand Computation Framework for Spatial Information Processing. In: GCCW 2006. Fifth International Conference on Grid and Cooperative Computing Workshops, Hunan, China, October 21-23, 2006, pp. 487–490

[8] http://en.wikipedia.org/wiki/Maximum_likelihood