Journal of Computer Science

Original Research Paper

Performance Evaluation of Apache Spark Vs MPI: A Practical Case Study on Twitter Sentiment Analysis

[1,2]Deepa S Kumar and [3,4]M Abdul Rahman

[1]Research Scholar, Karpagam Academy of Higher Education,
Karpagam University, Tamilnadu (Dist), Coimbatore, India
[2]Department of Computer Science and Engineering, College of Engineering Munnar, Kerala, India
[3]Pro-vice Chancellor, APJ Abdul Kalam Technological University, Kerala, India
[4]Research guide, Karpagam Academy of Higher Education,
Karpagam University, Tamilnadu(Dist), Coimbatore, India

Article history
Received: 12-10-2017
Revised: 25-11-2017
Accepted: 23-12-2017

Corresponding Author:
Deepa S Kumar
Department of Computer Science and Engineering,
College of Engineering
Munnar, Kerala, India
Tel: +91 9447524328
Email: deepa@cemunnar.ac.in

Abstract: The advent of various processing frameworks which happens under big data technologies is due to tremendous dataset size and its complexity. The speed of execution was much higher with High Performance computing frameworks rather than big data processing frameworks. As majority of the jobs under big data are mostly data intensive rather than computation intensive, the High Performance Computing paradigms were not been used in big data processing. This paper reviews two distributed and parallel computing frameworks: Apache Spark and MPI. Sentiment analysis on twitter data is chosen as a test case application for benchmarking and implemented on Scala programming for spark processing and in C++ for MPI. Experiments were conducted on Google cloud virtual machines for three data set sizes, 100 GB, 500 GB and 1 TB to compare the execution times. Results shown that MPI outperforms Apache Spark in parallel and distributed cluster computing environments and hence the higher performance of MPI can be exploited in big data applications for improving speedups.

Keywords: Big Data, High Performance Computing, Apache Spark, MPI, Sentiment Analysis, Scala Programming, Cluster Computing

Introduction

Processing of huge volume of data in a variety of forms is one of the major challenges in the last two decades. Several parallel computing architectures were involved in shared memory, multi processor, multi-core processing, distributed processing, cluster computing, massively parallel processing, grid computing and specialized parallel computers like FPGA implementation etc based on the level of hardware support (Sliwinski and Kang, 2017). The parallel computers are designed for High Performance Computing (HPC). In contrast, another type of parallelism can be observed in High Throughput Computing (HTC). HPC provides computational resources for working with large datasets and mainly focus on how fast the computations can be done in parallel by exhibiting high computing power in a very short span of time. Whereas HTC is looking for how many tasks can be completed over a long period of time.

Over the last few decades, processing of tremendous volume of data and its analytics have become one of the big challenges in the field of computing which leads to the concept of Big data and the associated processing. In data limited problems, data transfer time is more significant than processing time and in computationally limited problems, processing is faster than data transfer time (Sliwinski and Kang, 2017). Hadoop integrates processing and data and introduces application-level scheduling to facilitate heterogeneous application workloads and high cluster utilization (Jha *et al*., 2014). Hadoop implementation of Big data is dealing with the data-intensive task execution, with the data storage and job scheduling. Hadoop Map reduce, Apache Spark, Storm etc are the big data processing frameworks. Among the frameworks, Spark is faster compared to Map reduce for batch processing.

Paper compared and evaluated execution times of existing big data processing framework, Apache Spark and the High Performance Computing Library, MPI. The