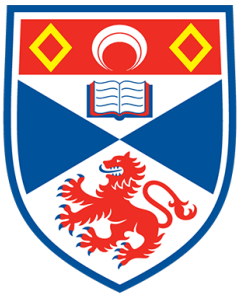
CS5052 – Paper Review

220031985



University of
St Andrews

03rd February 2023

"MapReduce: Simplified Data Processing on Large Clusters"

by Jeffrey Dean and Sanjay Ghemawat

1) INTRODUCTION

The presented work addresses the implementation of the MapReduce programming model, which is used to process and produce large data sets utilized in real-life applications. The underlying runtime system automatically parallelizes the computation across large clusters of machines, manages machine failures, and sets up inter-machine communication to make the most of the network and discs.

2) OBJECTIVE

To calculate many other types of derived data, including inverted indices, various graph representations of web documents, summaries of the number of pages crawled per host and the set of most popular searches on a particular day. The majority of these computations are theoretically easy. However, for the computations **to be completed in an acceptable amount of time**, the input data is often huge and dispersed across hundreds or thousands of workstations.

The problem was how to distribute the data, parallelise the process, and handle failures combine to conceal the initial straightforward algorithm with a lot of complicated code to address these problems.

The **team created a new abstraction to solve this complexity** which lets the user describe simple computations that they are attempting to carry out while encapsulating the messy specifics of parallelization, fault tolerance, data distribution and load balancing in a library.

The abstraction is modelled after the *"map"* and *"reduce"* primitives present in Lisp and other functional languages. The team discovered that the majority of their calculations required computing a series of intermediate *key/value pairs* and applying a **reduction operation** to all values that share the same key to appropriately combine the derived data. The use of a functional model with a user-specified map helps reduce operations allowing to parallelise large computations easily and to re-execute as a primary mechanism for fault tolerance.

3) CONTRIBUTION

The main contribution of this work was the design of a simple and efficient interface that enables *automatic parallelization* and *distribution of large-scale computation* while achieving great performance on massive clusters of commodity PCs, as well as the programming model's ability to utilise a system's multiple cores for parallel processing.

4) NOVELTY

The work is considered to be novel, as it provides a simple and effective solution to a previously challenging problem of parallel data processing on large clusters of computers.

It can be viewed as both an application of an existing solution to a new setting and a brand-new solution. They have used the idea of parallel processing to handle huge datasets on massive computer clusters, and the technique they suggest is unique in how easily it can be scaled up and down. A strong evaluation of the solution demonstrates MapReduce's efficiency in handling huge datasets.

The authors argued fact before MapReduce was in existence the existing systems were ineffective and hence proposed the solution. It has been extensively used in a wide range of domains within Google, as stated:

- Large-scale machine learning problems.
- Clustering problems for Google News and Google shopping (formerly Froogle) products.
- Data extraction to produce reports of popular queries.
- Processing Satellite imagery data
- Large – Scale graph computations

The most significant use of MapReduce to data has been a complete rewrite of the production indexing system that produces the data structures used for the Google Web search service.

5) EVALUATION

The authors have measured the performance of MapReduce by cluster configuration, Grep program, and sorting by running two computations on a large cluster of machines. MapReduce has been so effective that it enables programmers to create simple programs that can run efficiently on a thousand machines in a shorter amount of time, dramatically accelerating the development and prototyping cycle. The results show that MapReduce is *effective* and *scalable.*

For programmers with little to no experience in distributed/parallel systems, it makes resource exploitation simple. Although algorithms may

accept serial access to the data each pass, it does have certain limitations, such as the difficulty in defining sophisticated calculations like graph processing in the context of machine learning, where repeated runs through the data are necessary. and the need for additional monitoring and debugging tools.

6) OPINION

The MapReduce model has been an effective solution for the problem of processing large amounts of data on large computer clusters. Through experiments and real-world applications, it has been proven to work, and the solution's evaluation demonstrates this.

We cannot deny the fact that the MapReduce model established the industry standard for processing and analysing big data sets on a cluster of computers with limited resources, even though it is no longer the method of choice after Google switched to Apache Mahout in 2014.

It has been widely employed in several real-world applications and in processing big datasets on massive computer clusters. The study is regarded as making a *substantial contribution* to the domain of big data processing overall.

7) QUESTIONS

Q> As described above, the MapReduce model has certain limitations, have the authors addressed them or planning to address them shortly?

Q> What has happened to MapReduce since the paper has been written?

Q> Can the MapReduce Model be applied in various other industries and is there any plans on making it more accessible for developers?

Q> It is stated outside the paper, that the MapReduce model is no longer used by Google, why so and why not?

Total Words: 886