ISSN: 2583-9888
DOI: doi.org/10.51219/JAIMLD/rajalakshmi-thiruthuraipondi-natarajan/463

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science
https://urfpublishers.com/journal/artificial-intelligence

Vol: 1 & Iss: 3 *Research Article*

Big Data Processing: Advanced MapReduce Algorithm

Binoy Kurikaparambil Revi*

Citation: Revi BK. Big Data Processing: Advanced MapReduce Algorithm. *J Artif Intell Mach Learn & Data Sci 2023*, 1(3), 2119-2122. DOI: doi.org/10.51219/JAIMLD/rajalakshmi-thiruthuraipondi-natarajan/463

Received: 03 August, 2023; **Accepted:** 18 August, 2023; **Published:** 20 August, 2023

***Corresponding author:** Binoy Kurikaparambil Revi, Independent Researcher, USA, E-mail: binoyrevi@live.com

ABSTRACT

The challenge of computation of large volumes of data within the realm of Big Data is a famous problem. One prevalent strategy for addressing this challenge has been using the MapReduce algorithm, which facilitates the parallel processing of broken-down datasets across distributed systems. However, it is crucial to recognize that data processing requirements have become increasingly complex in today's world. Many of these requirements include complex conditional aspects that can add layers of complexity to handling extensive datasets. Consequently, while the MapReduce4 algorithm's ability to process data in parallel remains valuable, its effectiveness can sometimes be constrained by the conditional processing of complex data.

The Advanced MapReduce Algorithm presents a promising avenue for enhancing computing efficiency. This approach allows for handling complex data operations and effectively addresses some limitations of traditional MapReduce algorithms. By minimizing processing time, the algorithm builds upon its predecessor's strengths and aligns more closely with the sophisticated needs of conditional data processing challenges.

Keywords: Big data, Cloud computing, Map reduce, Data processing

1. Introduction

The MapReduce algorithm uses a simple and effective methodology within a computing cluster[3]. It harnesses the power of distributed computing by breaking down enormous datasets into smaller, manageable chunks. These smaller datasets are then processed across multiple computers (nodes) in the cluster. The designated node later consolidates the individual results obtained from these nodes into a single final result. However, the algorithm's efficiency may decline when processing involves specific conditions. This can occur in scenarios where a large dataset must be selected from a pool of several massive datasets based on the results of processing another large dataset. In such cases, the MapReduce algorithm must complete the first computation before it can proceed to the second part, which involves selecting the dataset based on the results of the initial computation.

The Advanced MapReduce Algorithm is highly beneficial in this context. This algorithm assigns the initial computation to a specific set of nodes rather than utilizing all available nodes. It then leverages the remaining nodes to begin parallel processing the second part of the computation. The algorithm selects a few datasets that are likely to be chosen using the result of the first computation or it may use all available datasets, depending on the node availability within the cluster[3] and the size of the datasets. This approach fully utilizes the computing cluster's power to solve the computational problem more quickly than merely using the MapReduce Algorithm.

2. Map Reduce Algorithm

The MapReduce algorithm is an excellent technique in cluster computing. It divides large datasets into smaller subsets for processing, allowing different nodes in the cluster to compute these subsets simultaneously. This approach enhances efficiency and speeds up data processing. Figure 1 demonstrates how the MapReduce algorithm works.

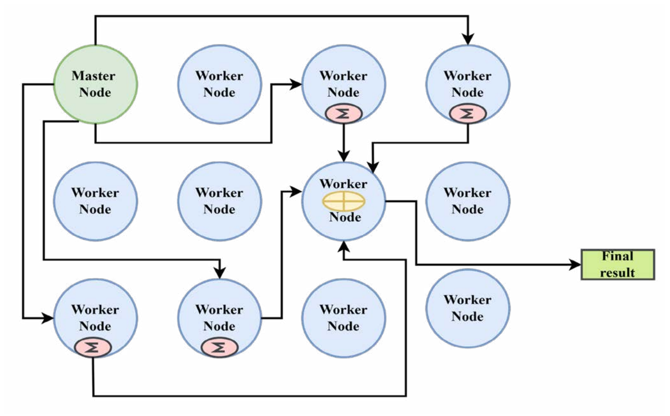

Figure 1: MapReduce Algorithm.

(Figure 1) shows that the Master Node divides the computation into smaller datasets and distributes them to four selected nodes for processing. This process is known as mapping. After the nodes complete the mapping function, another node gathers and processes the results through the reduce function. Let's represent this in an algorithm.

Consider an example of counting the number of one keyword in a large collection of documents[1]. The algorithm shall look something like this:

```
map(String key, String val)
        Int count = 0
For each word w in val
        Increment count if w == key
end
reduce(Array count)
        Int sum = 0
        For each val in count
                sum = sum + val
end
```

This approach works well with large datasets that follow a uniform processing pattern. However, when dealing with large datasets that require conditional processing, the computing speed of the MapReduce algorithm may decrease. This slowdown occurs because the processing of specific datasets may depend on the completion of other datasets. In such cases, MapReduce can provide significant assistance.

3. Advanced MapReduce Algorithm

(Figure 2) illustrates how the Advanced MapReduce Algorithm[4] operates. In addition to the Master node and worker nodes, this algorithm includes an intelligent node that employs a data-specific algorithm or machine learning model to determine which dataset to select for conditional operations.

As shown in Figure 2, there are 2 major parallel processing tasks are represented in black and orange lines.

The first task (marked in black) involves initial data processing. This task divides the large dataset into smaller datasets and supplies them to the four worker nodes for computations (represented in green). After the computations are completed, the results are used to calculate the intermediate results by the worker node executing the reduce function.

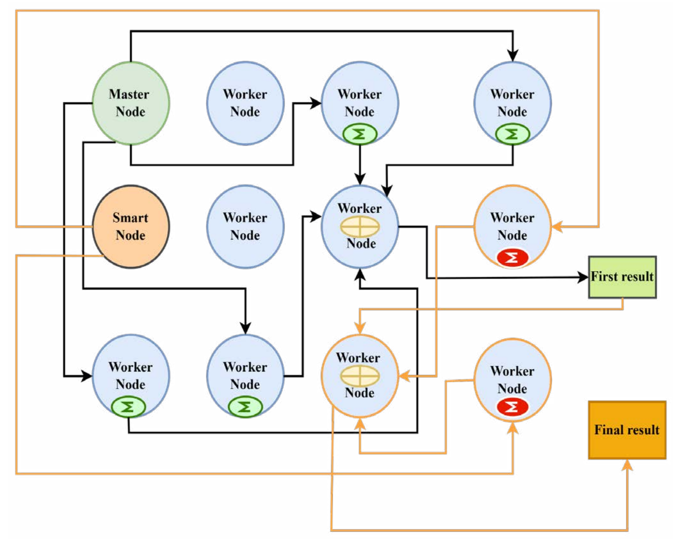

Figure 2: Advanced MapReduce Algorithm.

The second major task (highlighted in orange) is initiated simultaneously with the first task by the Smart Node. The Smart Node selects one or more datasets that are potential candidates for processing under complex processing conditions. It then divides the chosen datasets and supplies them to another set of computation nodes for processing. The results from these nodes, along with intermediate results from Task 1, are used by a worker node to derive the final outcome through a final reduce function.

Advanced MapReduce Algorithm looks like this if we can extend the previous example[1]:

```
T1map(String key, String val)
        Int T1count = 0
For each word w in val
        Increment T1count if w == key
end
T1reduce(Array T1count )
        Int T1sum = 0
        For each T1val in T1count
                T1sum = T1sum + T1val
end
T2map(String key1, String value)
        Int T2count= 0
For each word w in value
        Increment T2count if w == key1
end
reduce(Array T2count, T1sum )
        Int T2sum = 0
        For each T2val in T2count
                T2sum = T2sum + T2val
        final result = MLProcess(T1sum, T2sum)
end
```

4. Introducing Smart Node in Advanced MapReduce Algorithm

The Smart Node plays a crucial role in the Advanced MapReduce algorithm. It is designed to use algorithms that identify datasets based on specific application criteria and can be implemented more efficiently using AI or machine learning models. These models can predict which datasets will be needed for processing task 2. This allows task 2 to begin concurrently with task 1, facilitating the effective parallel processing of specific datasets under complex conditions.

5. Challenges with the Advanced MapReduce Algorithm

Advanced MapReduce Algorithm is undoubtedly an intelligent and effective technique for processing datasets under complex conditions. However, it also has some challenges, which are stated below:

Complex conditions and large datasets may necessitate using a large number of nodes, as the cluster[3] creates an execution plan to process Task 1 and Task 2 in parallel. Therefore, the increased demand for more nodes in the Advanced MapReduce Algorithm is unsurprising.

Sometimes, the model or algorithms may identify multiple datasets that qualify for Task 2 processing. When this occurs, the Smart Node loads all eligible datasets onto the worker nodes to begin processing Task 2. This approach can be costly, as the results from some of the nodes may not contribute to the final result.

6. Conclusion

The Advanced MapReduce Algorithm significantly enhances the conventional MapReduce framework, particularly addressing the intricacies associated with complex conditional data processing. Such complexities can considerably diminish the algorithm's operational efficiency, resulting in increased latency.

At the core of this advanced algorithm is the Smart Node, an intelligent component that employs predictive algorithms or trained artificial intelligence or machine learning models. The Smart Node plays a crucial role by facilitating predictive decision-making for conditional operations within the data processing sequence. By anticipating the outcomes of various conditions, the Smart Node allows for simultaneous processing alongside initial tasks, optimizing resource utilization and accelerating overall execution timelines. This ability for parallel processing enhances performance and significantly increases the algorithm's capacity to manage large and complex datasets, ultimately leading to improved effectiveness in challenging data environments.

7. References

1. https://dl.acm.org/doi/10.1145/1327452.1327492
2. https://www.sciencedirect.com/science/article/abs/pii/0167819188901093?via%3Dihub
3. Baker Mark and Rajkumar Buyya. "Cluster computing at a glance." High Performance Cluster Computing: Architectures and Systems.
4. Alam A and Ahmed J. "Hadoop Architecture and Its Issues," 2014 International Conference on Computational Science and Computational Intelligence, Las Vegas, NV, USA, 2014: 288-291.