T63_Study_Report.pdf

by Sumit Vishwakarma

Performance and Scalability of Broadcast Mechanisms in Spark

Course: IE494

Student ID: 202201320

Team ID: T63

Name: Sumit Vishwakarma

1) Introduction:

As the technology has advanced, the reach of online platforms has expanded exponentially. More the number of people have access to online platforms, more the information or data is generated. During the 2000s, the world storage size, as cited by rivery.io, was 54.5 Exabyte. And till the time of 2020s, it has increased to over 6800 Exabyte. This gives an approximate idea of the growth and increasing importance of data in our daily lives. One of the main tasks is the proper and efficient flow of these data, so they can be utilized efficiently to produce meaningful information.

1.1 A look into some frameworks

MapReduce and Apache Spark are few computing frameworks which helps to process a large volume of datasets (even work well with data up to petabytes).

Based on the technological data, it is evident that a single computer(node) cannot handle the processing of such large datasets alone. Clusters (a collection of independent machines) handles the processing of such large datasets as a single entity.

Certain algorithms like ALS algorithm are CPU intensive and some are data intensive which require the same set of data to work upon, like machine leaning algorithms involving linear and logistic regression, as well as page rank computation algorithms that works upon the same datasets until graph converges.

The programming model of Spark favors the processing of datasets in these types of algorithms.
It should be noted that if each machine works on the same dataset, they must have access to it locally. If large datasets are to be transferred over the network frequently, they will definitely lead to latency and network congestion, which leaves the process as inefficient.
This is where **Broadcast Mechanism** comes into the picture. Broadcasting works by the sharing of large datasets only once to all the executor nodes. This mechanism plays a vital role in distributing the datasets, so that the data processing can be occurred parallelly.

The main requirements of broadcast mechanisms are Performance, Scalability, Fault tolerance and topology independence.

2) Understanding of the study carried out:

Various broadcast mechanisms were thoroughly studied, involving Centralized HDFS broadcast (CHB), Chained Streaming Broadcast (CSB), BitTorrent Broadcast (BTB) and Split Stream Broadcast (SSB). Out of these BTB and SSB were experimental.

For a good broadcast mechanism, the broadcast time plot should be as close to constant (linear) as the number of executor nodes in cluster increases.

2.1) The workings of CHB mechanism involved saving the broadcast variables in the shred file system. If the executor node wants to access the file, spark checked the file in the cache, and if not present, then fetching it from the nearest replica (in order to minimize delay and maximize the performance).
Performance and Scalability: The data transferring rate reached maximum of 160 Mbps which is considered satisfactory. The results indicate that performance is compromised when the number of the executor nodes crossed 40 mark (as seen from the graph below).

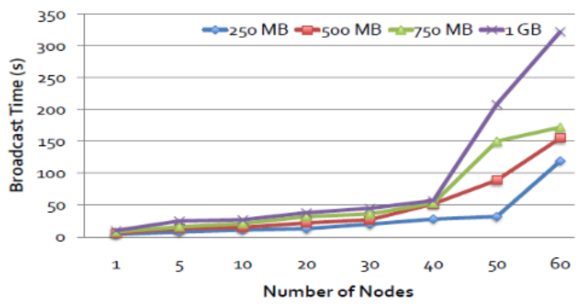

Figure 5: Scalability and performance of CHB (

Understanding: This mechanism is suitable for low to medium scaled cluster systems which require maximum 35-40 executor nodes.

2.2) The workings of CSB are different from CHB as it involves breakdown of the variable into multiple blocks. Any node which has even a fraction of the variable is present in priority queue, so that they can share it with the executor nodes. In this way the distribution occurs which is often similar to chained structure (involving a chain between a tracker and a slave). This method proved to scale much better the CHB.

Performance and Scalability: The results indicate that performance is comparable with the CHB mechanism until forty executor nodes, after which the CSB mechanism seems to scale better as compared to CHB mechanism.

Understanding: This method involves a concept similar to load balancing, in which the broadcasting files are unevenly distributed among multiple slave nodes.

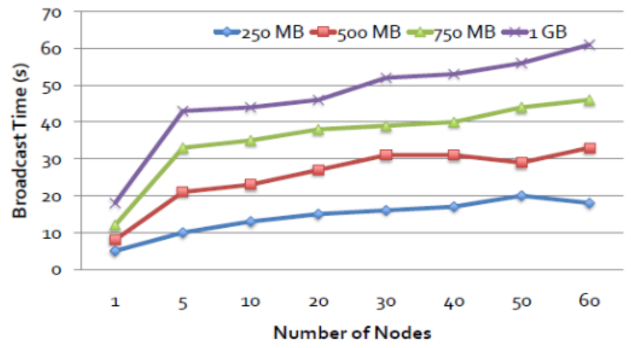

Figure 7: Scalability and performance of CSB (All nodes: m1.large EC2 instances).

2.3) The working model of BTB involves a concept similar to peer-to-peer file transfer on Internet. The data blocks are distributed in the .torrent files format.

Performance and Scalability: As compared to CSB and CHB, its performance is compromised due to the fact that multiple receivers can leave the session in between, introducing unnecessary delays.

Understanding: This mechanism is experimental and the fact that the mechanism is not scalable above 30 executor nodes have no explanation yet. This suggests it is only suitable for low level cluster systems but has a potential for high scalability.

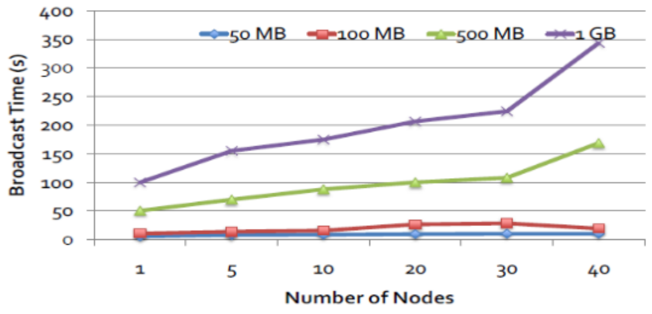

Figure 8: Scalability and performance of BitTorrent with default settings (All nodes: m1.xlarge EC2 instances).

2.4) The working of SSB model involves multicasting of variables after splitting them into multiple stripes so that the executor nodes can parallelly access any part of it. This method certainly improves efficiency. All nodes in this mechanism are willing to receive and further distribute the divided stripes.

Performance and Scalability: This mechanism is proved to be efficient for audio and video streaming. Some encoding schemes are required for fault tolerance.

3) Conclusions drawn: The following table concludes all the observations

The following table concludes all the observations:

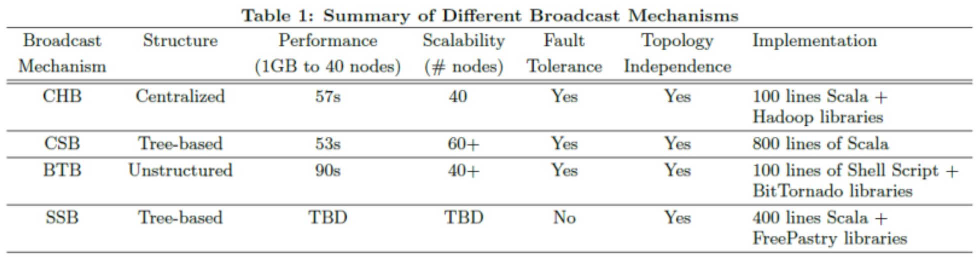

Table 1: Summary of Different Broadcast Mechanisms

Broadcast Mechanism	Structure	Performance (1GB to 40 nodes)	Scalability (# nodes)	Fault Tolerance	Topology Independence	Implementation
CHB	Centralized	57s	40	Yes	Yes	100 lines Scala + Hadoop libraries
CSB	Tree-based	53s	60+	Yes	Yes	800 lines of Scala
BTB	Unstructured	90s	40+	Yes	Yes	100 lines of Shell Script + BitTornado libraries
SSB	Tree-based	TBD	TBD	No	Yes	400 lines Scala + FreePastry libraries

On the basis of the experiments carried out, the most efficient broadcast mechanism in terms of performance and scalability is the Chained Streaming Broadcast (CSB). As the technologies and online platforms are evolving, it is clear that in the near future, the amount of data generated and to be processed will be exponentially increased as compared to now. In that case, efficient data processing would be a mandatory requirement to successfully handle such large datasets. This will eventually lead to high level cluster system involving multiple nodes (up to 100 and more) to parallelly complete the task as a single entity. CSB is proved to be helpful but there's still a scope of improving the broadcast mechanisms to achieve high scalability.

4) Ideas for future study/work based the report

As mentioned in the report that two of the mechanisms i.e. BitTorrent and Split Stream Broadcast were experimental. In this way in the near future, there is scope of modelling a new broadcast mechanism which would identify the drawbacks of these aforementioned broadcast mechanisms to improve the performance in terms of latency, load balancing among nodes and throughput. Moreover, with the advent of 6G and other upcoming technologies, we can model and suitably experiment mechanisms to handle low latency and very high bandwidth data transfer scenarios. Also, in the near future, software models and systems may require algorithms (like ML algorithms in present day) that are data intensive. In that case, it would be crucial to develop such broadcasting mechanism that could scale better as the volume of data increase.

5) References:

[1] Chowdhury, Mosharaf, Matei Zaharia, and Ion Stoica. “Performance and scalability of broadcast in Spark.” press. 2014.

[2] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: high-bandwidth multicast in cooperative environments.

[3] Educational blog: Optimizing PySpark: Broadcast Variables and Broadcast Joins (Medium.com)

T63_Study_Report.pdf

ORIGINALITY REPORT

0%	0%	0%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		