Reduce Cost in Relational Domain Using a New Progressive Analytics

Ankita V. Torawane **Prof. Amitkumar Manekar**

***Abstract* — The term Progressive analytics means to reduce the cost of data analytics in relational domain. The samples of increasing data size or progressive samples uses sampling strategies for exploratory querying. User-control, repeatable semantics, and result provenance are provided by the sampling strategies. Hence this type of solutions results in tedious workflows. Existing query processing systems gives results, where the above benefits discussed are not offered for complex ad-hoc queries. To avoid this a new progressive analytics system is proposed which is based on a progress model called Prism which allows users to communicate progressive samples to the system; also allows efficient and deterministic query processing over samples and provides repeatable semantics and provenance to data scientists. The "progress aware merge", algorithm is applied to the prism which consist key components like batching, data shuffle, merge and progress aware reducer. Using the application of Prism model with MR which is applied to another computational model where the results are represented in the graphical form and also the cost of the relational domain will be reduce. PageRank is implemented using the Hadoop MapReduce framework for calculating the PageRank value.**

***KeyWords*- Progressive Analytics, Approximate Query Processing (AQP), Map reduce online**

I.INTRODUCTION

The increase in volume of data storage and process in the cloud, where analytics on such type of data is very expensive day by day. The cloud causes computation cost to increase with query execution time which makes it possible for data scientist to spend large amount of money on analyzing the data. Traditionally query must be execute to completion before problems are diagnosed. Scientist choose to perform querying on extracted samples of data The approach gives them control to choose from huge variety of sampling strategies in specific manner. Given sample provides precise query semantics, result provenance and repeatable execution of query. Choose a fixed samples size for all queries and operate over multiple progressive samples of increasing size.

Approximate Query Processing (AQP) performs the progressive analytics which is proposed by the database community systems which are DBO [14]. Progressive analytics produce results of analytics queries based on partial data and the results are refined as more data is received. Sufficient accuracy or query incorrectness is observed in the results of progressive analytics. Map reduce online(MRO)[12] adds pipelining to MR but MRO reports the progress metrics that does not eliminate the problems related to map reduce.

The lack of system support results in tedious and error prone work that precludes the reuse of work across progressive samples.The system is needed which allows user to communicate with progressive samples to the system and allow efficient and deterministic query processing samples. The key idea is for users to encode their chosen progressive sampling strategy into the data by augmenting tuples with explicit progress intervals (PIs). PIs denote logical points where tuples enter and exit the computation, and explicitly assign tuples to progressive samples. PIs offer flexibility forencoding sampling strategies and ordering for results, including arbitrarily overlapping sample sequences.

A new progress model is introduce into an existing relational engine appears challenging. A progressive in-memory relational engine based on Prism can be realized immediately using an unmodified temporal streaming engine, to denote the progress reuse of temporal fields are carefully done. Tuples from successive progressive samples getincrementally processed when possible, giving a significant performancebenefit. The temporal engine is unaware thatit is processing atemporalrelational queries; the queries can simply re-interpretits temporal fields to denote progress points. While it may appearthat in-memory queries can be memory intensive since the finalanswer is computed over the entire dataset, Prism allows us to exploitsort orders and foreign key dependencies in the input data andqueries to reduce memory usage significantly.

AQP is generalized by prism progress semantics which are compatible with queries for which prior AQP techniques with statistical guarantees apply, and thus don't require user involvement. These techniques simply correspond to different PI assignment policies for input data. Variants of ripple join [18] are different PI assignments for a temporal symmetric-hash-join, with intervals computed as part of the query. Thus, Prism is orthogonal and can leverage this rich area of prior work, while adding the benefits of repeatable and deterministic semantics. Prism gives progressive results form of determinism and control the final results.

The Prism model is particularly suitable for progressive analytics on big data in the Cloud, since queries in this setting are complex, and memory and CPU intensive. Scalable distributed frameworks such as MR are not pipelined, making them unsuitable for progressive analytics. MRO[12] adds pipelining, but does not offer the semantic underpinnings of progress necessary to achieve the desirable features.

II.LITERATURE REVIEW

Online aggregation is proposed by Hellerstein et al. [20], where the focus was on grouped aggregation with robust confidence intervals based on random sampling. This was again extended to handle join queries using the ripple join operators[19]. Laptev [7] proposed MR jobs to compute iteratively on increasing data samples until a desired approximation goal is achieved. BlinkDB [1] constructs a

large number of multi-dimensional samples online using a particular sampling technique ie stratified sampling and choose samples based on a user-specified budget. Instead of taking systems responsibility for query accuracy (e.g., as sampling techniques) which may not be possible in general is followed with a different approach, which involves the query writer's specification of progress semantics. A query processor using Prism model support a variety of user-defined progressive samplingschemes; that helps the assignment of PIs in a semantically appropriate manner.

Map-Reduce Online (MRO) [12] supports progressive output by adding pipelining to MR. The result of snapshots are produced by reducers, each annotated with a rough progress estimate based on averaging progress scores from different map tasks.Progress in MRO is an operational and non-deterministic metric that cannot be controlled by users or used to correlate progress to query accuracy or to specific input samples. MRO sorts subsets of data by key and redundant computations as reducers repeat aggregations over increasing subsets [5].

Li et al. [8] propose scalable one pass analytics (SOPA), where he replace sort-merge in MR with a hash based grouping mechanism inside the framework. The focus of the paper is on progressive queries, with a goal of establishing and propagating explicit progress in the platform. Like SOPA, the framework sorting is eliminated to leave it to the reducer to process progress-sync ordered data. Stream engines use efficient hash-based grouping, which allows to realize similar performance gains as SOPA inside our reducers.

Progressive results for atemporal queries over atemporal online data, and show that new progress model can in fact be realized by leveraging and re-interpreting the notion of time used by temporal SPEs. Now! is an MR-style distributed framework for progressive queries; marked in different from distributed SPEs [15] as it leverages the explicit notion of progress to build a batched-sequential data-parallel framework that does not target real-time data or low-latency queries.

Dremel [11] and PowerDrill [2] are distributed system for interactive analysis of read-only large columnar datasets. Spark [3] provides in-memory data structures to persist intermediate results in memory, and is used to interactively query big data sets or get medium-latency results on real-time data . These engines have a different goal they provide full results to queries in-memory data in milliseconds, for which they use careful techniques such as columnar in-memory data organization for the (smaller) subset of data that needs such interactivity. The generic interactivity is processed over large datasets, in terms of meaningful results on progressive samples and refining results. Based on results, users can choose to potentially end (or possibly refine) computations sufficient accuracy or query incorrectness.

In online aggregation [9], a database system processes a user's aggregation query in an online fashion. An estimate of the final query result is given to the system while processing. The paper focus on the idea of online aggregation which can be built into a MapReduce system for large-scale data processing.

C.Jermaine [14] describes query processing in the DBO database system. The other database systems are designed for ad-hoc, analytic processing, DBO is able to compute the exact answer to queries over a large relational database in a scalable fashion.DBO constantly maintain a guess as to the final answer to an aggregate query throughout execution, along with statistically meaningful bounds for the guess's accuracy which are designed for analytic processing. As DBO gathers more and more information, the guess gets more and more accurate, until it is 100% accurate as the query is completed. This allows users to stop the execution at any time that they are happy with the query accuracy, and encourages exploratory data analysis.

B.Chandramouli [6] focuses the concept of "Big Data" in map-reduce (M-R) clusters isfundamentallytemporal. Display advertising uses BehaviouralTargeting (BT) to select ads for users based on prior searches,page views, etc. Previous work on BT has focused on techniquesthat scale well for offline data using M-R. There limitations for BT-style applications that deal with temporal data many queries are temporal and not easily expressible in M-R, and moreover, the set-oriented nature of M-R frontends such as SCOPE is not suitable for temporal processing; and commercial systems mature, they may need to analyze and react to real-time data feeds since a high turnaround time can result in missed opportunities, but it is difficult forcurrent solutions to operate over real-time streams.

M.Hammad focus on concept NILE [16]or streamInsight[13] which can modify a database engine to add PI support to all operators in the engine. The idea is to leverage a stream processing engine (SPE) as the progressive query processor. The semantics underlying a temporal SPE such as NILE can be leveraged to denote progress, with the added benefit of incremental processing across samples when possible. Nile extends the query processor engine of an object-relational database management system to support data streams.

P.Haas and J.Hellerstein[18,19] present join algorithms, which are also called ripple joins, for online processing of multi-table aggregation queries in a relational database management system. Such queries arise naturally in interactive exploratory decision-support applications. Online join algorithms are designed to minimize the time to completion of the query. Whereas ripple joins are designed to minimize the time until an precise estimate of the query result is available, measured by the length of a confidence interval. Ripple joins are adaptive, adjusting their behavior during processing in accordance with the statistical properties of the data. Ripple joins also permit the user to dynamically trade off the two key performance factors of online aggregation. The time between successive updates of the running aggregate, and the amount by which the confidence-interval length decreases at each update. How ripple joins can be implemented in an existing dbms using iterators are shown, and an overview of the methods used to compute confidence intervals and to adaptively optimize the ripple join parameters.

III.METHODOLOY

The prism model architecture depends on system architecture which is based on the map reduce (MR) computation paradigm. The overall design is compared with MapReduce for query with two stages and different

partitioning keys. User in the figure indicate the input and output data on distributed cloud storage and can be replaced by any distributed storage such as HDFS[5].

The key points are as follows :-

3.1 Progress aware data flow
The prism progress model is implemented which support for data flow in strictly progress sync order. The data flow of progress batches are governed by the PI's(progress interval). The main components of progress aware data flow are-

3.1.1 Progress aware batching
The input data is partitioned into number of input splits, data tuples which are assigned progress intervals in progress sync order. Mapper reads input data as progressive samples which invokes users map function. The resulting key value pairs are partitioned by key to produce progress batches with same progress sync value and which has unique ID. Input text reader appends an end of file (eof) when it reaches the end its input data which appends it to all progress batches sequence.

3.1.2 Progressive data shuffle
Shuffles the data between the mapper and the reducer of progress batches without sorting. Fine grained signaling mechanism pipelines progress batch from mapper to reducer which allows the mapper to inform the job tracker. The job tracker passes the batch ID and location information to reducer where the download mechanism is designed to support progress sync ordered batch movement. A separate blocking concurrent queue (BCQ) is maintain by reducer for each mapper associated with the job. The maximum size of the BCQ is set according to the available memory of reducer. Reducer enqueues progress batches download from each mapper into corresponding BCQ in strict progress sync order.

3.1.3 Progress aware merge
The prism model uses progress aware merge mechanism which ensures data flow in progress sync order along all paths in the framework. Input taken by the algorithm are number of mapper M, set of BCQ B where qi € B denotes the blocking concurrent queue of mapper I, current progress sync value Cmin of merged batch and H where hi € H is progress sync value currently at head of qi. Initialize an empty set O as output, iterates mapper queue to find and dequeue batches whose progress sync value match Cmin, add them to O and hi to new value head of qi. Update Cmin and return O, merged batch having same sync value. O is the progressive reducer. If O=ø indicates end of input on BCQ, framework passes eof to progressive reducer signaling termination.

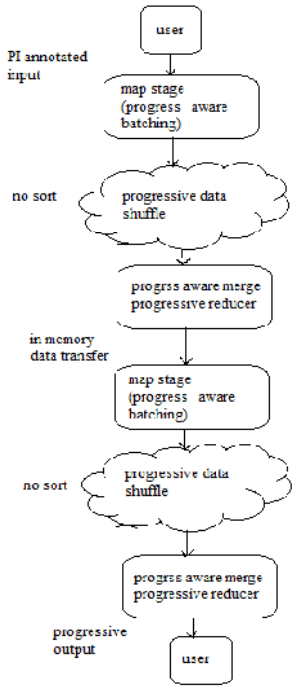

Figure 1:- System Architecture[5]

3.1.4 Progress aware reducer
In MapReduce the reducer gathers all the values for each key and invokes reduce function for each key, passing the values associated with the keys.Prism uses progress aware reducer

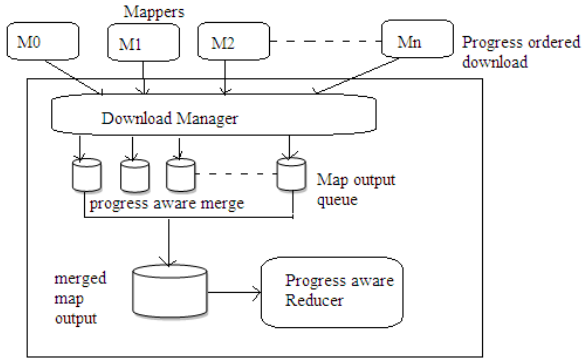

Figure 2:-Progress aware merge

where the input is arranged in sequence of progress batches associated with partition in progress sync order. Per key grouping and computation and produce a sequence of batches in sync order as output is the responsibility of the reducer.
The above figure 1 & 2 illustrate the system architecture and progress aware merge which describe the three method used for map reduce technique.

3.2 Implementation Details

Cloud computing programming model widely use Map-Reduce class , in which map functions read in the input data, and emits multiple intermediate <key, value> pairs. Reduce

functions then merge the emitted <key, value> pairs such that all values associated with the same key are paired together. Programs written using map and reduce functions are automatically parallelized and scheduled on a large cluster of commodity machines. In this process, scheduling, balancing maintenance, and error detection and recovery are automatically managed by the cloud computing platform such as Hadoop. Users only need to concern with the implementation detail of the algorithm.

For scalable progressive analytics the Hadoop MapReduce framework is considered which use the MapReduce class to propose a new progress prism model.The progress aware merge data flow is applied consisting the key points which are batching data shuffle and progress merge.

For the PageRank value the Wikipedia has 3.7M articles at the moment and is still growing where each article is link with other articles. With those incoming and outgoing links we can determine which page is more important than others, which basically is what Page Ranking does. The use following Wikipedia article xml dump is used for PageRank computation of each page with is of 5.5 GB in size.

Experiment setup will be considered out on different configuration of number of nodes to form Hadoop cluster. Individual nodes capacity is of 2 cores, 2 Gb RAM, 500 Gb hard disk space. With Hadoop version 2.6.0 supporting Ubuntu 14.04 operating system.

IV.MATHEMATICAL MODEL

For performing the mathematical module consider the example of an advertising platform where computing the click through rate (CTR) for each ad is analyze. Two sub queriesQc and Qi which compute number of clicks and impression are required. Qc needs to process clicks on per user basis which consider clicks from webpage, Qi need to process different set of logged data.in final query Qctr merge the results of Qc and Qi, and compute the ratio as CTR.

Input:-Click data(ad data)

Output:-Qc , Qi and Qctr (click-through-rate)

CD-click data
Z=(CD,S,f1,f2,f3,f4)

Function f1: (CD)→(S)
Where,
CD-click data(CD1,CD2,…….CDn);
S-set of key value pairs (K,V)
Transformation of click data into list of key value pairs.

Function f2: (S)→(Qc)
Computation of Qc,
Qc(K,V)→MR;
//using Map Reduce program

Function f3: (Qc,S)→Qi
Computation of Qi,
Qi,Qc(K,V)→MR;
//using Map Reduce program

Function f4:(Qc,Qi,S)→Qctr
Computation of Qctr,
Qctr,Qi,Qc(K,V)→MR;
//using Map Reduce program

V.PERFOMANCE ANALYSIS

The expected result for the experiment is to reduce the cost of relational domain by proposing a new progress model called prism model. The results are represented in the graphical form with reduce cost of relational domain. With the help of the progressive samples the approach uses the clicks dataset to compute CTR progressively. Hadoop MapReduce framework calculates the PageRank value considering the two important factors efficiency and effectiveness. MapReduce process the entire input in single batch and the batch size controls number of progress batches.

The following table shows the time considered with the help of multinodes attach with the reversible cables in a synchronize manner. Progress aware computation of PageRank on Hadoop defines the time reduce with the help of PageRank computation using the MapReduce framework. Using the PageRank computation time is reduce using the concept of multinodes. Considering number of nodes according to the size of RAM time is calculated with progress aware and without progress aware computation. This computation is useful for the computation like PageRank which shows the Progress aware computation.

No .of nodes	Memory capacity (RAM)	Time for PageRank Computation (secs)	Time for Progressive PageRank Computation (secs)
1	2 GB	667secs	608secs
3	6GB	575secs	525secs
6	12GB	500secs	467secs

Table1 :-Progress Aware Computation of PageRank on Hadoop

The above table1 clearly shows the time which is reduce in terms of cost considering the progressive analytics.For node 1 which has 2GB RAM time consider for PageRank computation is 667secs and with progressive PageRank is 608secs were time is reduce viceversa.similar for 6GB and 12GB RAM the tmie reduce is 525secs and 467secs respectively.thus shown time is reduce in terms of cost considering progressive PageRank computation.

Barchart is shown for better undersatanding of the progressive aware computation of PageRank value.The Progressive analytics shows the time reduce in terms of cost.

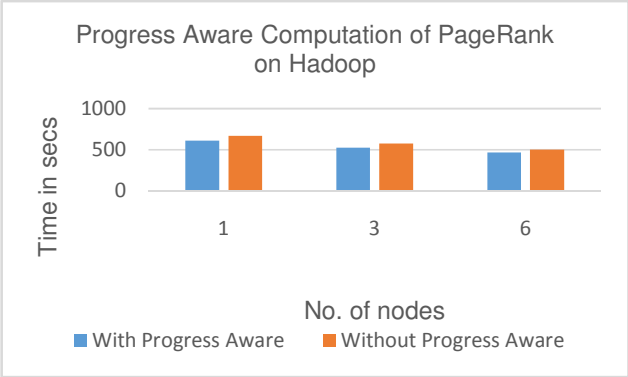

Figure 3:-Progress aware computation chart

VI. Conclusion

A new progress model called Prism a new progress model is proposed which allows users to communicate progressive samples to the system;allows efficient and deterministic query processing; and provides repeatable semantics and provenance to the data scientists.Prism model apply map reduce to another computational model where the results are represented in the graphical form and also the cost of the relational domain will be reduce. PageRank value is also calculated using the PageRank class for Hadoop MapReduce framework.

Acknowledgment

The authors would like to acknowledge Computer Engineeringdepartment, SITRC and all the people who providedwith the facilities being required and conductive conditionsfor completion of the paper.

References

[1] S. Agarwal et al. Blinkdb: Queries with bounded errors and bounded response times on very large data. In EuroSys, 2013.
[2]A. Hall, O. Bachmann, R. B¨ussow, S. G˘anceanu, and M. Nunkesser.Processing a trillion cells per mouse click. PVLDB July 2012.
[3] M. Zaharia et al. Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. NSDI'12.
[4] M. Zaharia et al. Discretized streams: An efficient and fault-tolerant model for stream processing on large clusters. In HotCloud, 2012
[5]B. Chandramouli et al. Scalable progressive analytics on big data in Thecloud. Technical report, MSR.2012
[6]B. Chandramouli et al. Temporal analytics on big data for web advertising. In ICDE, 2012.
[7] N. Laptev, K. Zeng, and C. Zaniolo. Early accurate results for Advanced analytics on mapreduce. PVLDB 2012.
[8] B. Li, E. Mazur, Y. Diao, A. McGregor, and P. J. Shenoy. A Platformfor scalable one-pass analytics using mapreduce. In SIGMOD 2011
[9] N. Pansare, V. R. Borkar, C. Jermaine, and T. Condie. Online aggregation for large mapreduce jobs. PVLDB, 2011.
[10] P. Upadhyaya, Y. Kwon, and M. Balazinska. A latency and fault-tolerance optimizer for online parallel query plans. In SIGMOD, 2011
[11] S. Melnik et al. Dremel: interactive analysis of web-scale datasets. PVLDB 2010
[12] T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, K.Elmeleegy, and R. Sears. Mapreduce online. In NSDI, 2010.
[13] M. Ali et al. Microsoft CEP Server and Online Behavioral Targeting. 2009.
[14] C. Jermaine, S. Arumugam, A. Pol, and A. Dobra. Scalable approximate query processing with the dbo engine. SIGMOD'07.
[15] D. Abadi et al. The design of the Borealis stream processing engine.2005.
[16]M. Hammad et al. Nile: A query processing engine for data streams.2004.
[17] J. Dean and S. Ghemawat. Mapreduce: simplified data processing Onlarge clusters. OSDI'04.
[18] P. J. Haas and J. M. Hellerstein. Ripple joins for online aggregation.InSIGMOD 1999.
[19]P. J. Haas and J. M. Hellerstein. Join algorithms for online aggregation. In IBM Research Report RJ 10126, 1998.
[20]J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. InSIGMOD, 1997.

Author's Profile

Author'Name : Ankita V. Torawane
B.E. Computer(Pune University)
PG Student,
SavitribaiPhule Pune University,
SITRC College,
Nashik-422213
E-mail id: ankita1891@gmail.com

Author's Name: Asst. Prof. Amitkumar Manekar
Work Experience 1.6 Yrs in Company and 5 Yrs in teaching Ph.D Pursuing in CSEas specialization in Big Data Analytic and Cloud Computing Domain . Manekar A.S. Malti Nalge, et al."Sifting Of A Potent Convex Hull Algorithm For Scattered Point Set Using Parallel Programming", 2013 5th International Conference on Computational Intelligence and Communication Networks,IEEE/D OI 10.1109/CICN.2013.121.