iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 Issue: VII Month of publication: July 2022

DOI: https://doi.org/10.22214/ijraset.2022.45880

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue VII July 2022- Available at www.ijraset.com

To Build and Optimize ETL Pipeline

Prakhar Kant Kushwaha[1], Prof. Swetha S[2]
[1, 2]Information Science Engineering, RV College of Engineering, Bengaluru, India

Abstract: As the world is getting digitized the speed in which the amount of data is over owing from different sources in different format,there is a need for an Environment for better processing and optimization of Data also called as Big Data.Building and optimization of ETL pipeline facilitates the loading and refreshment of Data warehouse contents which is required by other Teams for Prediction and Analysation purposes .This Project demonstrates ETL Pipeline of Predicting TV Tune-In and churn rate for a particular user , various optimization techniques were discussed and performed in order to decrease cost and increase Performances of the computation Processes.

Keywords: DSPs , ACR , Spark , Databricks , Extraction , Transform , Load

I. INTRODUCTION

Companies required big processing technologies to research the huge amount of information. They use Big Data technologies to come back up with Predictions to cut back the danger of failure and to be able to optimise that data in line with the requirements of various Teams working like Data Science and BA Teams.Building and optimization of ETL pipeline facilitates the loading and refreshment of information warehouse contents which is required by other Teams for Prediction and Analysation purposes .This Project demonstrates ETL Pipeline of Predicting TV Tune-In and churn rate for a selected user , various optimization techniques were discussed and performed so as to decrease cost and increase Performances of the computation Processes.Modern ETL process tools break down data and provides self-service capabilities to the those that understand the information best, letting them draw more informed conclusions from one source of truth in less time.

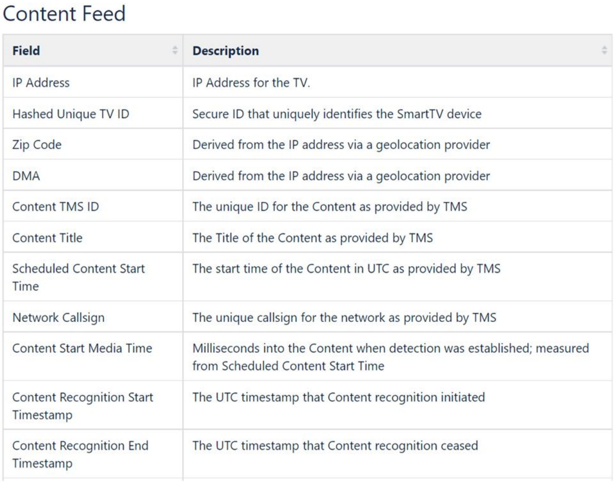
II. LITERATURE SURVEY

Marketing Intelligence and Big Data by [1] Prof. Dr. habil. Jan Lie FOM University of Applied Science, Dortmund (Germany) shows vast scope of digital application areas, which shape the digital marketing landscape and coin the present term “marketing intelligence” from a marketing technique point of view. Additionally, marketing intelligence as social engineering techniques are described. [2]Big Data Optimization Techniques: A Survey by Chandrima RoySiddharth Swarup Rautaray and Manjusha Pandey Kiit University, Bhubaneswar, India In this paper various optimization techniques has been presented using big data tools. It is also esteemed the existing optimization technologies, mechanism and techniques of big data framework gives a summary of several methods and highlights most of the substantial outcomes of existing research.[3]Big data analytics on Apache Spark Salloum, S., Dautov, R., Chen, X, In this paper, review on the key features of Apache Spark for big data analytics provides a variety of functionalities for designing, implementing and tuning machine learning algorithms and pipelines.[4]Programmatic trading: the future of audience economics by Dan Andrew , The News and Media Research Centre, University of Canberra, Canberra, Australia This paper examines how audiences are commoditised in the audience marketplace by commercial media providers and advertisers have been using audience economic models that were applied to commercial television, radio and print audiences.[14] touches upon efficient heuristics for logical optimization of the ETL workflows. [12] extended the ETL optimization to physical implementation for the logical counterparts. Recently metrics other than cost have also been considered for optimization. [18] introduces the idea of optimizing ETL workflows for quality metrics such as reliability and scalability. [16] considers optimizing ETL workflows for external interruptions like faults etc. All these ETL optimization strategies assume the availability of statistics necessary in determining the cost of the operator and focus on the process of cost-based optimization using the operator’s cost. In contrast, the primary contribution of our work is to address the issue of estimating the operator cost when the input statistics are either missing or incomplete.

III. METHODOLOGY

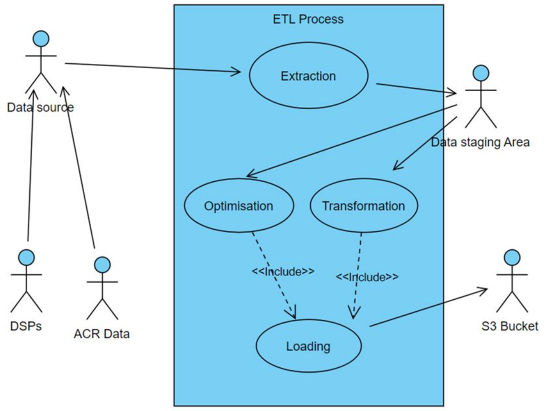
In the first stage the objective of the project is identified i.e to extract data , transform and optimize it and finally loading of data. Here all the data are provided by various DSPs (Demand side platform) like Appnexus , Trade Desk along with ACR (Automatic content recognition) data provider , The dataset contains attributes like IP Address , Unique TV Id , zipcode , start_timestamp etc. all these data is collected and extracted into a staging area in AWS S3 bucket lakehouse structure .

After which the data is transformed according to the Requirements provided by the Data science team , operations like join , groupBy were applied and optimised SQL queries were written and processed inside the Spark engine using AWS EC2 machine which is the computation engine, Bottlenecks were identified in order to increase the efficiency of the whole Data.

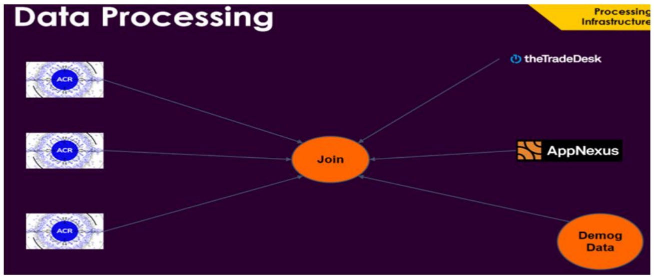


Content Feed

Field	Description
IP Address	IP Address for the TV.
Hashed Unique TV ID	Secure ID that uniquely identifies the SmartTV device
Zip Code	Derived from the IP address via a geolocation provider
DMA	Derived from the IP address via a geolocation provider
Content TMS ID	The unique ID for the Content as provided by TMS
Content Title	The Title of the Content as provided by TMS
Scheduled Content Start Time	The start time of the Content in UTC as provided by TMS
Network Callsign	The unique callsign for the network as provided by TMS
Content Start Media Time	Milliseconds into the Content when detection was established; measured from Scheduled Content Start Time
Content Recognition Start Timestamp	The UTC timestamp that Content recognition initiated
Content Recognition End Timestamp	The UTC timestamp that Content recognition ceased

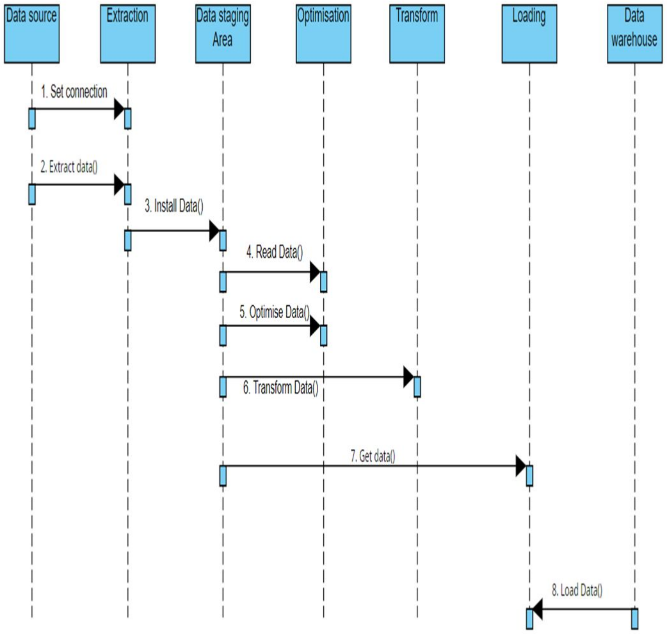


Finally after all the operation were performed the optimised data is loaded back to the Amazon S3 bucket were further it will be used for Analysis and Prediction Purpose.



The Data processing involves multiple joins and the aggregation of data coming from DSPs and ACR data. The data is join with demographic data for a particular user who is watching the television all these are combined in such a way so that it can be further used for finding user for the targeting and also used for preparing test data for data modelling and prediction.Apache Spark is a large data analytics engine that is free and open source. It is capable of handling batch and real-time analytics and data processing workloads.

It is based on Hadoop MapReduce and extends the MapReduce architecture to allow it to be used efficiently for a wider range of calculations, such as interactive queries and stream processing. Python has native Spark bindings. It also comes with numerous libraries to help you construct machine learning [MLLib], stream processing [Spark Streaming] and other applications. Spark Core and a group of libraries make up Apache Spark. Spark Core is the brains of Apache Spark, and it's responsible for distributed task transmission, scheduling, including I/O. The Resilient Distributed Dataset (RDD) is the basic data type used by the Spark Core engine. The RDD is built in such a way that it hides the majority of the computing complexity. Data and partitions are pooled over a server cluster, where they may then be computed and can either relocated to a separate data store or processed via an analytic model, thanks to Spark's intelligence.



IV. BIG DATA OPTIMISATION

There are lots of optimization that were applied into the data to make the pipeline more efficient which are as follows, Firstly the requirements were thoroughly studied and analyzed only after which data is processed and rolled according to those requirements which reduces the data size considerably.The scripts and stages in spark jobs were analysed in order to find out processing bottlenecks and it was found that there are very few python UDFs that were being used all those were converted into spark UDFs and couple of UDFs were using pandas for ETL operation these were converted back to spark DFs which was causing an overhead on the driver ,in this way all Pandas operations were converted into spark operations which caused parallelism in multiple stages of the job to go up.

Since there were large datasets join involved in pipeline, joins were found out to be slower in the initial stages. To improve this Skew Hint is used to improve Join performance.Since the ETL operation was a long one, problems like long DAGs arrived and any intermediate failure lead to start from scratch for backfill. For avoiding it intermediate caching & persisting DFs in S3 as checkpoints were implemented.And finally it was observed that the initial configuration selected for the job was not ideal as it has lots of unused memory. The job required more executors compared to memory and hence it is switched to C series which has double the executors compared to r series and with 40% lesser price.

V. SOFTWARE TESTING

A. Unit testing of modules

Unit test is the verification effort on the smallest unit of software design, the software modules. Unit testing ensures that the bugs that occur can be pinpointed easily since the code tested on is a small unit. The section describes some of the unit tests run with test case details and brief explanations.

Sl.No.	1
Test Case	test _rawinput
Feature being Tested	ACR and DSPs Data being stored as an external dataset in AWS Data Lake
Description	Batches of incoming data has to be correctly partitioned into RDDs and store in AWS S3 Buckets
Sample Input	Unstructured data with multiple key values/ structured data with multiple columns
Expected Output	RDDs partitioned evenly without skew
Actual Output	RDDs partitioned evenly without skew
Remarks	The Raw ACR and DSPs data has been successfully partitioned into RDDs and stored in S3 bucket.

Table describes the test case which evaluates the script written for the data Extraction process. On verifying whether the right structure of data has been Extracted the data is partitioned into RDDs.

B. Integration Testing

Integration Test Cases are distinct from other test cases in that they concentrate on the interface and flow of data/information between modules. The integrating connections take precedence over the unit functions, which have previously been tested. Following individual module testing, the integration of these modules is checked for correctness. Integration Test Cases are distinct from other test cases in that they concentrate on the interface and flow of data/information between modules.

The integrating connections take precedence over the unit functions, which have previously been tested. Following individual module testing, the integration of these modules is checked for correctness.

Sl.No.	2
Test Case	test_storageverification
Feature being Tested	Post Transformation and optimisation, ensuring loading of data
Description	Once transformations and optimisation are done, data needs to be loaded in Amazon S3 bucket data warehouses or sent to synapse for business analytics. Testing needs to be done on this module to ensure data is reaching the endpoint
Sample Input	RDDs which have been shuffled and flat mapped
Expected Output	Successful verification status code
Actual Output	Successful verification status code
Remarks	The correct data flow is observed in this test.

Table 5.4 describes the evaluation of the data loading module once the RDDs have undergone the required transformations and need to be stored or used for further analytics.

C. System Testing

System testing is the process of combining all of the modules that have been tested through integration testing to create a single system. The system is tested to ensure that all of the units are properly interconnected to meet the needs of the users. This test aids in the removal of overall bugs and improves the system's quality and assurance. System testing determines the system's correct functionality. This system testing evaluates the entire system, including all of the major modules. The system testing is as shown in Table.

Sl.No.	3
Test Case	test_fullverification
Feature being Tested	Full Functionality Check
Description	Full data flow has to be verified
Sample Input	Unstructured data with multiple key values/ structured data with multiple columns
Expected Output	Successful Data Storage or Data Mapping to AWS Bucket (Request’s response and AWS Monitoring)
Actual Output	Successful Data Storage or Data Mapping to AWS Bucket
Remarks	The correct data-flow is observed in this test.

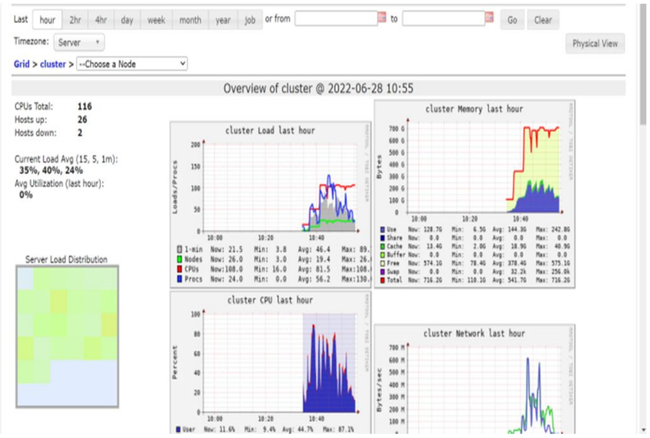
Table shows Successful test Case for Data Processing and Loading .

Sl.No.	4
Test Case	test_AWS
Feature being Tested	AWS Computation processing check
Description	Full data flow has to be verified
Sample Input	Optimized script written in SQL
Expected Output	Proper utilization of memory with no executors getting killed
Actual Output	Unused memory executors getting killed
Remarks	The correct data-flow is observed in this test.

Table shows failed test case where because of wrong configuration executors are getting killed which can be handle by switching to C series which has double the executors in R series

VI. RESULTS & ANALYSIS

DSP and ACR data is successfully extracted and loaded into the S3 bucket , Data is successfully Transformed and optimized according to the Requirements and finally Data is successfully loaded into the warehouse with 35 % Improvement in cost and 20% Improvement in Performance.



Overview of cluster @ 2022-06-28 10:55

CPUs Total: 116
Hosts up: 26
Hosts down: 2

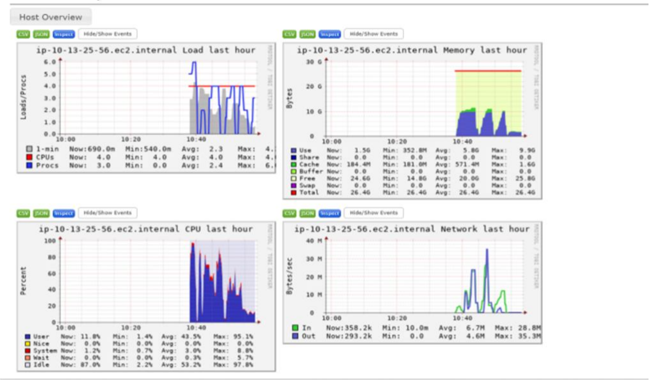
Current Load Avg (15, 5, 1m):
35%, 40%, 24%
Avg Utilization (last hour):
0%

```
1-min   Now: 21.5   Min:  3.8   Avg: 46.4   Max: 89.
Nodes   Now: 26.0   Min:  3.0   Avg: 19.4   Max: 26.
CPUs    Now:108.0   Min: 16.0   Avg: 81.5   Max:108.
Procs   Now: 24.0   Min:  0.0   Avg: 56.2   Max:130.
```

```
Use     Now: 128.7G   Min:   6.5G   Avg: 144.3G   Max: 242.8G
Share   Now:   0.0    Min:   0.0    Avg:   0.0    Max:   0.0
Cache   Now:  13.4G   Min:   2.0G   Avg:  18.9G   Max:  40.9G
Buffer  Now:   0.0    Min:   0.0    Avg:   0.0    Max:   0.0
Free    Now: 574.1G   Min:  78.4G   Avg: 378.4G   Max: 575.1G
Swap    Now:   0.0    Min:   0.0    Avg:  32.2k   Max: 256.0k
Total   Now: 716.2G   Min: 110.1G   Avg: 541.7G   Max: 716.2G
```

```
User  Now: 11.6%   Min: 9.4%   Avg: 44.7%   Max: 87.1%
```

The whole ETL process of TV tune-In and churn pipeline, optimization of its Data with 35 % Improvement in cost and 20% Improvement in Performance were implemented and how it can be used for other similar pipelines. The spark jobs created for the ETL process can be further optimized and Transformed using more optimized SQL queries ,that may lead to decrease in cost of computation on using AWS engine and an increase in Performance of ETL process i.e less time and more throughput.If the performance of a system running traditional computational methodologies and a system in a cluster running Spark were to be compared, then the latter is 100x faster. This is because of the advantage of local computing and Parallel processing. Hence Spark is prevailing everywhere, where the big data is to be analyzed. One of the reasons which caused the formulation of the objectives is to have a decoupled model where processing is isolated from the storage.


Host Overview
CSV
JSON
Inspect
Hide/Show Events
ip-10-13-25-56.ec2.internal Load last hour
Loads/Procs
6.0
5.0
4.0
3.0
2.0
1.0
0.0
10:00
10:20
10:40
1-min Now:690.0m Min:540.0m Avg: 2.3 Max: 4.
CPUs Now: 4.0 Min: 4.0 Avg: 4.0 Max: 4.
Procs Now: 3.0 Min: 0.0 Avg: 2.4 Max: 6.
ip-10-13-25-56.ec2.internal Memory last hour
Bytes
30 G
20 G
10 G
0
Use Now: 1.5G Min: 352.8M Avg: 5.8G Max: 9.9G
Share Now: 0.0 Min: 0.0 Avg: 0.0 Max: 0.0
Cache Now: 184.4M Min: 181.0M Avg: 571.4M Max: 1.6G
Buffer Now: 0.0 Min: 0.0 Avg: 0.0 Max: 0.0
Free Now: 24.6G Min: 14.8G Avg: 20.0G Max: 25.8G
Swap Now: 0.0 Min: 0.0 Avg: 0.0 Max: 0.0
Total Now: 26.4G Min: 26.4G Avg: 26.4G Max: 26.4G
ip-10-13-25-56.ec2.internal CPU last hour
Percent
100
80
60
40
20
0
User Now: 11.8% Min: 1.4% Avg: 43.5% Max: 95.1%
Nice Now: 0.0% Min: 0.0% Avg: 0.0% Max: 0.0%
System Now: 1.2% Min: 0.7% Avg: 3.0% Max: 8.8%
Wait Now: 0.0% Min: 0.0% Avg: 0.3% Max: 5.7%
Idle Now: 87.0% Min: 2.2% Avg: 53.2% Max: 97.8%
ip-10-13-25-56.ec2.internal Network last hour
Bytes/sec
40 M
30 M
20 M
10 M
0
In Now:358.2k Min: 10.0m Avg: 6.7M Max: 28.8M
Out Now:293.2k Min: 0.0 Avg: 4.6M Max: 35.3M

This would serve the purpose of keeping the servers which serve the apps and where the data is generated, alive. Before Spark, the data was analyzed locally in those legacy servers and the servers had to be shut down temporarily while the analysis was in progress as the computational power demanded by the analytics engine wouldn't suffice. Spark entirely eliminates the need to shut down by having the data migrated to the cluster and processing it locally. Another advantage of Spark is in the form of time preservation. Before Spark, developers had to keep staring at the screen until the job was done. Developers, especially from the Data chapter, were as less productive as they could be due to it. Switching to Spark has bought the company a huge deal of developing time and has eliminated almost 95% of the waiting time of the job execution.

VII. CONCLUSION

In this Project, ETL pipeline is built and Optimize, for Predicting TV tune-In and churn rate. Data is Extracted from ACR Technology and various DSPs like AppNexus. Data Extracted were Transformed according to the Requirements for Data Science Team using Pyspark and AWS cloud Engine. Data went through various optimisation Techniques which led to a 35 % Improvement in cost and 20% Improvement in Performance. Finally Data is loaded into the S3 bucket from where it can be used for Analysis , Visualisation and Prediction etc.The ETL operation tasks were successful in achieving the required results and do not have any limitations per se but more experimentation can be done in optimizing the performance even more by reducing the storage as well as the computational costs that are incurred during the course of the execution of the Spark jobs on AWS Databricks.The spark jobs created for the ETL process can be further optimized and Transformed using more optimized SQL queries ,which will lead to decrease in cost of computation on using AWS engine and an increase in Performance of ETL process i.e less time and more throughput.

VIII. ACKNOWLEDGEMENT

I am indebted to my guide, Prof. Swetha S,Assistant Professor,Dept of Information Science and Engineering for her wholehearted support, suggestions and invaluable advice throughout our project work and also helped in the preparation of this thesis. I would also like to extend my gratitude to Bitanshu Das, Team Lead DE MIQ Digital India for his guidance in the completion of the project. I also express our gratitude to our Panel members for their valuable comments and suggestions. My sincere thanks to Dr. Sagar B M, Professor and Head, Department of Information Science and Engineering, RVCE for his support and encouragement. I express sincere gratitude to my Principal, Dr. K. N. Subramanyam for his motivation towards this project work.I thank all the teaching staff and technical staff of the Information Science and Engineering department, RVCE for their help. Lastly, I take this opportunity to thank our family members and friends who provided all the backup support throughout the project work.

REFERENCES

[1] Mr. Marisiddanagouda. M, M. R. M. Survey on performance of hadoop map-reduce optimization methods.International Journal of Recent Research in Mathematics Computer Science and Information Technology 2 (2021),114-121.

[2] Salloum, S., Dautov, R., Chen, X. et al. Big data analytics on Apache Spark. Int J Data Sci Anal **1,** 145–164 (2016). https://doi.org/10.1007/s41060-016-0027-9

[3] Awan, A.J., Brorsson, M., Vlassov, V., Ayguadé, E.: Architectural impact on performance of in- memory data analytics: Apache spark case study. CoRR arXiv:1604.08484 (2021)

[4] M. Bala, D. Verma (2018), "A Critical Review of Digital Marketing", International Journal of Management, IT & Engineering, Vol. 8 Issue 10,October 2020, pp. 321–339.

[5] Agarwal, R. Agrawal, R. Khanna, and N. Kota. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2010), pages 213--222, 2020

[6] Bhattacharya, M., Islam, R., and Abawajy, J. Evolutionary optimization: a big data perspective. Journal of network and computer applications 59 (2016), 416-426.

[7] Dong, B., Zheng, Q., Tian, F., Chao, K.-M., Ma, R., and Anane, R. An optimized approach for storing and accessing small files on cloud storage. Journal of Network and Computer Applications 35, 6 (2012), 1847-1862.

[8] Gu, R., Yang, X., Yan, J., Sun, Y., Wang, B., Yuan, C., and Huang, Y. Shadoop: Improving mapreduce performance by optimizing job execution mechanism in hadoop clusters. Journal of parallel and distributed computing 74, 3 (2014), 2166-2179.

[9] Hua, X., Wu, H., Li, Z., and Ren, S. Enhancing throughput of the hadoop distributed file system for interaction-intensive tasks. Journal of Parallel and Distributed Computing 74, 8 (2014), 2770-2779.

[10] Kolomvatsos, K., Anagnostopoulos, C., S. An efficient time optimized scheme for progressive analytics in big data. Big Data Research 2, 4 (2015), 155-165.

[11] Mr. Marisiddanagouda. M, M. R. M. Survey on performance of hadoop map-reduce optimization methods. International Journal of Recent Research in Mathematics Computer Science and Information Technology 2 (2015), 114-121.

[12] Nagina, D., and Dhingra, S. Scheduling algorithms in big data: A survey. Int.J. Eng. Comput. Sci 5, 8 (2016).

[13] [Nghiem, P. P., and Figueira, S. M. Towards efficient resource provisioning in mapreduce. Journal of Parallel and Distributed Computing 95 (2016), 29-41.

[14] Rumi, G., Colella, C., and Ardagna, D. Optimization techniques within the hadoop ecosystem: A survey. In Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2014 16th International Symposium on (2014), IEEE, pp. 437 444.

[15] Informatica. How to Achieve Flexible, Cost-effective Scalability and Performance through Pushdown Processing. Whitepaper, Nov. 2007.

[16] N. Kabra and D. J. DeWitt. Efficient Mid-Query Re-Optimization of Sub-Optimal Query Execution Plans. In L. M. Haas and A. Tiwary, editors, SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA, pages 106–117. ACM Press, 1998.

[17] R. Karp. Reducibility Among Combinatorial Problems. Complexity of Computer Computations, pages 85–103, 1972.

[18] N. Kumar and P. S. Kumar. An Efficient Heuristic for Logical Optimization of ETL Workflows. In BIRTE, pages 68–83, 2010.

[19] V. Markl, V. Raman, D. E. Simmen, G. M. Lohman, and H. Pirahesh. Robust Query Processing through Progressive Optimization. In SIGMOD Conference, pages 659–670, 2004.

[20] A. Simitsis. Mapping conceptual to logical models for ETL processes. In DOLAP, pages 67–76, 2005.

[21] A. Simitsis, P. Vassiliadis, and T. K. Sellis. State-Space Optimization of ETL workflows. IEEE Trans. Knowl. Data Eng., 17(10):1404–1419, 2005.

[22] A. Simitsis, K. Wilkinson, M. Castellanos, and U. Dayal. QoX-driven ETL design: reducing the cost of ETL consulting engagements. In SIGMOD Conference, pages 953–960, 2009.

[23] A. Simitsis, K. Wilkinson, U. Dayal, and M. Castellanos. Optimizing ETL workflows for fault-tolerance. In ICDE, pages 385–396, 2010.