IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

RAS: A PROFILING BASED DYNAMIC RESOURCE ALLOCATION FOR BIG DATA STREAMS

[1] VijayaKumar Chandarapu, [2] Madhavi Kasa

[1] Research Scholar, [2] Associate Professor

[1]Department of Computer Science and Engineering

[1]Jawaharlal Nehru Technological University College of Engineering,

[1]Jawaharlal Nehru Technological University, Ananthapuramu - 515002, Andhra Pradesh, India

[2] Department of Computer Science and Engineering

[2] Jawaharlal Nehru Technological University College of Engineering,

[2] Jawaharlal Nehru Technological University, Ananthapuramu - 515002, Andhra Pradesh, India

Abstract: Big data streams have become ubiquitous due to the fact that a number of applications generate a huge amount of data at a great velocity. This made it difficult for existing data mining tools, technologies, methods, and techniques to be applied directly on big data streams due to the inherent dynamic characteristics of big data.

The growing need to extract knowledge from big data streams has pioneered the challenge of selecting appropriate resources dynamically. Need for an approach that can efficiently overcome those drawbacks and to handle the stream data that takes into account of varied typed resources and application requirements integrally. The profiling of big data streams is a challenging task for getting insights. The profiling includes determining the types of data and dynamically allocating the resources.

Index Terms **- Big Data Streams, Profiling, Resources, Allocation.**

I. INTRODUCTION

Big data Stream Processing enables users to update with continuous data stream and early detection of required conditions within a small time period, where in Dynamic Resource provisioning for big data applications is a challenging issue for stream data analysis.
Stream processing is a technique that allows for the collection, integration, analysis, visualization, and system integration of stream data in real time, powering on-the-fly analytic that leads to immediately actionable insights.

The evolution of Big Data, especially its adoption by industry and government, expands the content/ meaning of Big Data[10]. The original volume-based definition now encompasses the data itself, relevant technologies and expertise to help generate, collect, store, manage, process, analyze, present and utilize data, as well as the information and knowledge derived.

However, there are three major problems in these resource allocation mechanisms. First, when a single application is running in the cluster with the default resource allocation mechanism, it will consume all the resources. As a consequence, resource sharing among applications will be prevented. Second, in static resource allocation, the user has to manually set the amount of resources each application is going to use. Even with dynamic resource allocation, the user still has to set the initial amount of resources[9]. As a result, improper allocation of resources might lead to severe performance issues. Lastly, if a production cluster has user-specific deadlines, default resource allocation mechanism may not work since any application with a strict deadline might have to wait in the FIFO queue. Furthermore, inappropriate resource allocation in both static and dynamic resource allocation techniques might affect the deadlines.

1.1 Objectives

Efficient two-layer scheduling approach to perform improved resource utilization in terms of latency, computation cost for streaming data based on cloud application. Two layers of scheduling comprises of partitioning and replication algorithm for streaming data resources to be allocated effectively[30].

Design the novel prediction-based resource allocation approach based on learning algorithm helps to predict the online streaming data usage through the process of metadata[23]. It helps to perform dynamic process to allocate the resource in appropriate datacenter and server the users with minimized delay.

Optimizing the resource allocation through modified optimization algorithm to perform effective scheduling of resource by balancing the load and energy effectively. The problem of load balancing andresource scheduling is considered to perform effective resource allocation with QoS in streaming data.

II. LITERATURE SURVEY

Past research on processing Big Data focused on the distributed and stream-based processing (Zikopoulos and Eaton 2011)[11]. While cloud computing emerged a bit earlier than Big Data, it is a new computing paradigm for delivering computation as a fifth utility (after water, electricity, gas and telephony) with the features of elasticity, pooled resources, on-demand access, self-service and pay-as-you-go (Mell and Grance 2011). Past research on processing Big Data focused on the distributed and stream-based processing (Zikopoulos and Eaton 2011)[6]. While cloud computing emerged a bit earlier than Big Data, it is a new computing paradigm for delivering computation as a fifth utility (after water, electricity, gas and telephony) with the features of elasticity, pooled resources, on-demand access, self-service and pay-as-you-go (Mell and Grance 2011).

Advanced sensors and their hosting devices (e.g. mobile phones, health monitors) are connected in a cyber-physical system to measure time and location of humans, movement of automobiles, vibration of machine, temperature, precipitation, humidity and chemical changes in the atmosphere (Lohr 2012)[8]. The Internet of Things (IoT, Camarinha-Matos, Tomic, and Graça 2013) captures this new domain and continuously generates data streams across the globe with geographical footprints from interconnected mobile devices, personal computers, sensors, RFID tags and cameras (Michael and Miller 2013; Van den Dam 2013). Astronomy is producing a spatiotemporal map of the universe (Dillon 2015) by observing the sky using advanced sky survey technologies.

Mapping and surveying the universe generates vast amounts of spatiotemporal data. For example, Sloan Digital Sky Survey (SDSS) generated 116 TB data for its Data Release 12 (Alam et al. 2015)[16]. New observational instruments (e.g. Large Synoptic Survey Telescope) scheduled for operation in 2023 will generate 15 TB data nightly and deliver 200 PB data in total to address the structure and evolution of the universe (http://www.lsst.org). In the fourth industrial revolution (Industry 4.0), products and production systems leverage IoT and Big Data to build ad-hoc networks for self-control and self-optimization (O'Donovan et al. 2015). Big Data poses a host of challenges to Industry 4.0[27].

2.1 Computational requirements

Parallel computing is one of the most widely used computing solutions to address the computational challenges of Big Data (Lin et al. 2013) through HPC cluster, supercomputer, or computing resources geographically distributed in different data centers (Huang and Yang 2011)[13,3]. To speed up the computation, this computing paradigm partitions a serial computation task into subtasks and uses multiple resources to run subtasks in parallel by leveraging different levels of parallelization from multi-cores, many cores, server rack, racks in a data center and globally shared infrastructure over the Internet (Liu 2013)[19].

The main purpose of literature survey is:

a) Identifying sources of Data
b) Identifying the types of resources
c) Identifying Storage of stream Data
d) Stream Processing Challenges
e) Identifying existing algorithms
f) Drawbacks of existing algorithms
g) Stream Data Load Prediction

There are many sources of streaming data such as

- Surveillance camera data,
- Twitter feed,
- a sensor network,
- connected IoT devices,
- buy-sell orders from a financial marketplace etc..

2.2 Resource Types

Resource describes any physical or virtual component of limited availability within a computer system[25].

- Infrastructural layer – Cluster: Shared Core to High-CPU and High-memory machines, Topology of Clusters
- Computational Layer – CPU, Memory, Disk, Bandwidth and Ports
- Specific Hardware – GPUs

2.3 Existing Solutions

- Navroop Kaur, sandeep sood "data stream analyzed on cloud to obtain output by workload estimator and estimating volume and variability " here the author described an estimator for volume ,velocity and variety to address the problem[5].
- R. Jordan Crouser, Lyndsey Franklin, Kris Cook "Because of the incoming data's overwhelming scale, sampling and filtering are a necessity".
- Rafael Tolosana,javier diaz " proposed queuing theory-based controller for provisioning computational resources"[14].
- Karim Kanoun, Cem Tekin "A scheduler is able to learn online to assign processing method to each stream and how to allocate its resources over time in order to maximize the performance".

2.4 The existing Resource allocation techniques:

Table 2.1 Resource allocation methods and Cons

S.No	Resource Allocation Techniques	Disadvantages
1	Memory Less resource Fairness (MLRF) Allocates resources without historic information.	Single Resource - Trivial Workload Problem
2	Long Term Resource Fairness (LTRF) Allocates resources based on past history.	Single Resource- Holding shared resources unnecessarily
3	Inter-tenant Resource Trading Allocates resources exactly according to the demand. Inter-tenant Weight Adjustment Allocates share of resources based on demand.	Multiple Resource - Ignores the imbalance of resource requirements (at different time periods)

Issues with Existing Schedulers

- Static assignment
- Not able to tackle runtime changes
- Require user's intervention to specify resource requirements
- Resource agnostic
- Mismatch of task resource demands and node resource availability
- Leading to over/under utilization

- Load balancing design
- Endeavour to distribute the workload as evenly as possible
- Unable to perform resource consolidation when the input is small

In-Memory Database

- An in-memory database (IMDB, also known as a main memory database or MMDB) is a database whose data is stored in main memory to facilitate faster response times.
- Source data is loaded into system memory in a compressed, non-relational format.
- In-memory databases streamline the work involved in processing queries [21].

Stream Data Load Prediction

To predict Stream Data Load, we used online support vector regression scheme.

Existing Models

a. Autoregressive Moving Average Model (ARMA),
b. Support vector regression.

Disadvantage: These methods are not real online predictions and cannot handle real-time stream data.

III. PROPOSED WORK FLOW

3.1 Dynamic Resource-efficient Scheduling

A user-transparent mechanism in Distributed Stream Management System(DSMS) to schedule streaming applications as compact as possible, matching the resource demands of streaming tasks to the capacity of distributed nodes, so that fewer resources are consumed to achieve the same performance target. This includes

- Profile streaming tasks at runtime to get accurate resource demands.
- Model and solve the scheduling problem as a bin-packing variant to enable resource consolidation.
- Dynamically allocate the resources.

The figure3.1 Shows the proposed base system, which takes data streams as input and gives to resource definer whose job is to categorize the incoming data and assign appropriate resources by using the Resource allocation scheme.

The cluster manager responsible for taking care of worker nodes, who actually does the given task. Workers are the physical/compute nodes where one or more application processes can be created depending on the resource capacity. In cloud deployments, one or more worker nodes can be created inside each allocated Virtual Machines (VM). A cluster can have one or more worker nodes but there is only a single Master node that is responsible for managing the worker nodes.

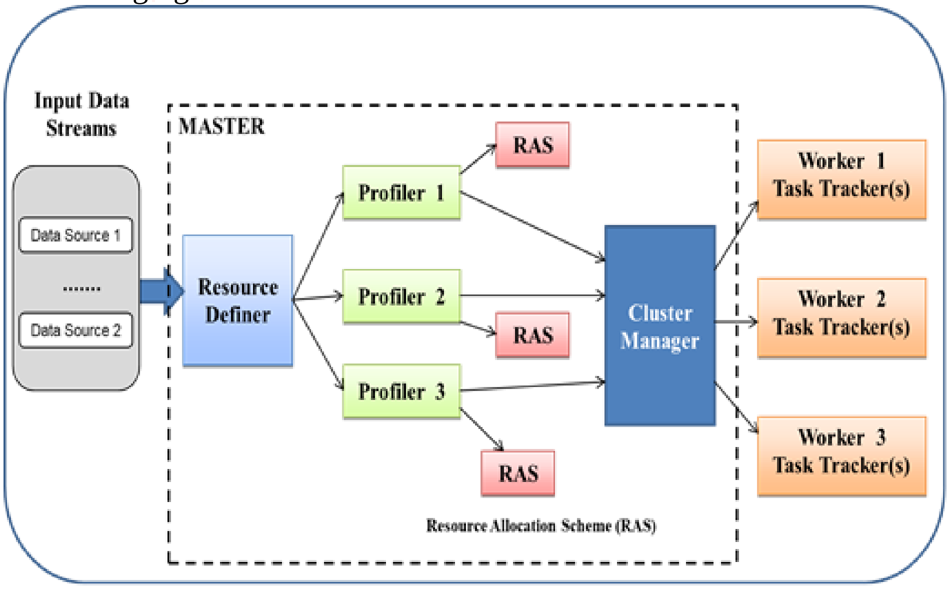


Figure 3.1 Resource Definer and Profiler

Notations:

n -The number of tasks to be assigned
Ti -Task i , i = 1,2….. n
m - The number of available worker nodes in the cluster
vi - Worker node i , i = 1,2…m
W c vi - CPU capacity of vi , measured in a point-based system
W m vi - Memory capacity of vi, measured in Mega Bytes (MB)
vused -The set of used worker nodes in the cluster
mused - The number of used worker nodes in the cluster
Pc - Unit CPU requirement for Ti to process a single tuple
Pm - Unit memory requirement for Ti to process a single tuple

The main contributions of this work are as follows:

- We design an automatic, light-weight, pluggable resource allocation framework that works from the master node along with the underlying cluster manager.
- We propose a resource allocation model where a cost-effective, deadline-based Resource Allocation Scheme (RAS) can be found for an application.
- We propose a model that predicts the completion time of an application based on the number of executors and properties of the application.
- We develop a Spark Profiler to profile any application with respect to varying input workloads, iterations, resource allocation schemes etc.
- We propose a simple algorithm to generate Resource Allocation Schemes (RAS) which can be used to deploy applications in an Apache Spark cluster.
- We implement the framework using the proposed models and algorithms. In addition, we run comprehensive experiments to show the accuracy and performance benefits of our proposed models.

IV. PROBLEM FORMULATION

Cost-efficient Resource Allocation Model- A cluster comprising of master and worker nodes can be deployed on cloud Virtual Machines (VM). For simplicity of our proposed model, we assume that all the VMs used as worker nodes are homogeneous. Therefore, each of the VM will have same amount of CPU cores, memory and storage disk. To deploy an application in the cluster, a Resource Allocation Scheme (RAS) needs to be defined. In each RAS, the total number of executors, CPU cores in each executor and memory in each executor should be specified. Our goal is to choose a cost-efficient RAS which ensures that an application will be completed before the user-specified deadline.

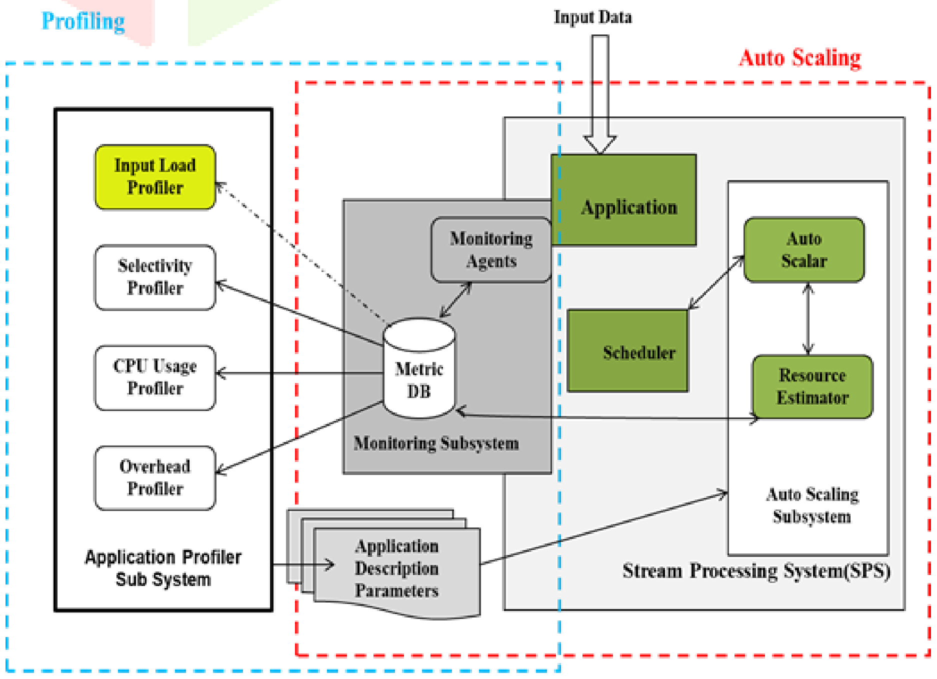


Figure 3.2 Auto Scaling and Stream Processing System

The application uses it's allocated executors to process multiple chunks/splits of the whole input in parallel. The partitioning or splitting of the input imposes a little overhead on the actual running time. In addition, after all the processing is finished, the result needs to be serialized which also adds up to the total execution time. If the number of input chunks is more than the number of executors, these input chunks are processed like a batch in each executor. However, adding too many executors to achieve more parallelism can cause overheads due to serialization, de-serialization and intensive shuffle operations in the network.

4.1 Stream Data Load Prediction

Online Support Vector Regression

- ➢ Can learn and update the data online.
- ➢ Gives different values to different time periods and improves the accuracy of data load prediction.
  1. Stream data is sent in time window '*t*'.
  2. Stream data is modelled as time series

$$X= \{x_1, x_2, x_3, . . . , x_n\}$$

  1. The past values are used to predict the future value *f(x)* and output.
  2. The prediction of stream data load in next time window '*t*' is

$$x_{n+t} = f\,(x_n, x_{n-1}, x_{n-2}, . . . , x_{n-m+1})$$

- ➢ The historical stream dataset is represented as

  $Z =\{(x_1, y_1), (x_2, y_2), . . . , (x_i, y_i)\}\ \varepsilon\ (X \times Y)^l$.

  X represents the time feature vector, and

  Y represents the stream data load feature vector.
- ➢ From the above we found Maximum Sustainable throughput of time window(MSTW): to deal with fluctuating stream data load.

Algorithm

Input: *A, T, R_{SLA}, $R_{(A,T)}$, F, $MSTW_{SLA}(m, T)$*

```
1) Begin
2) $R_{(A,T)}$ = r
3) if $F \geq MSTW_{SLA}(m, T)$
4) EXTDVMs ← min VMs - m
5) else if $F < MSTW_{SLA}(m, T)$
6)          if $r > R_{SLA}$
7)           EXTDVMs ← 1
8)          else
9)          CONTVMs ← m – min VMs
10) End
```

Output: EXTDVMs or CONTVMs

With the application A, the time window interval T, the response time limit R_{SLA}, the application average response time $R_{(A,\ T)}$, the payload prediction value F, distribution algorithm maximum sustainable throughput of time window (MSTW), m number of VMs, the output is the number of extended virtual machine EXTDVMs or the contractile virtual machine CONTVMs.

4.2 Resource Allocation

- Allocates resources to each query and picks its configuration based on the allocation.
- Has an offline profiler that efficiently generates the query's resource-quality profile for its different configurations, and an online scheduler jointly maximizes the quality and minimizes the lag of streaming queries.
- Given a profile P_k and a utility function U_k for each query k, the scheduler allocates resources A_k to the queries and picks their query configuration ($C_k \in P_k$).
- The scheduler runs periodically (e.g., every few seconds) and reacts to arrival of new queries, changes in query demand and changes in resource capacity (e.g., due to other high priority non-Video Storm jobs).

4.3 Experiment Setup

15 GPUs(Nvidia)

- 5 worker nodes: 2 VCPUs, 6GB RAM and 30GB disk
- 1 Nimbus, 1 Zookeeper, 1 Kestrel node
- Extended on Apache Storm v1.0.2, with Oracle JDK 8

Two test applications

- Twitter Sentiment Analysis data
- CCTV data

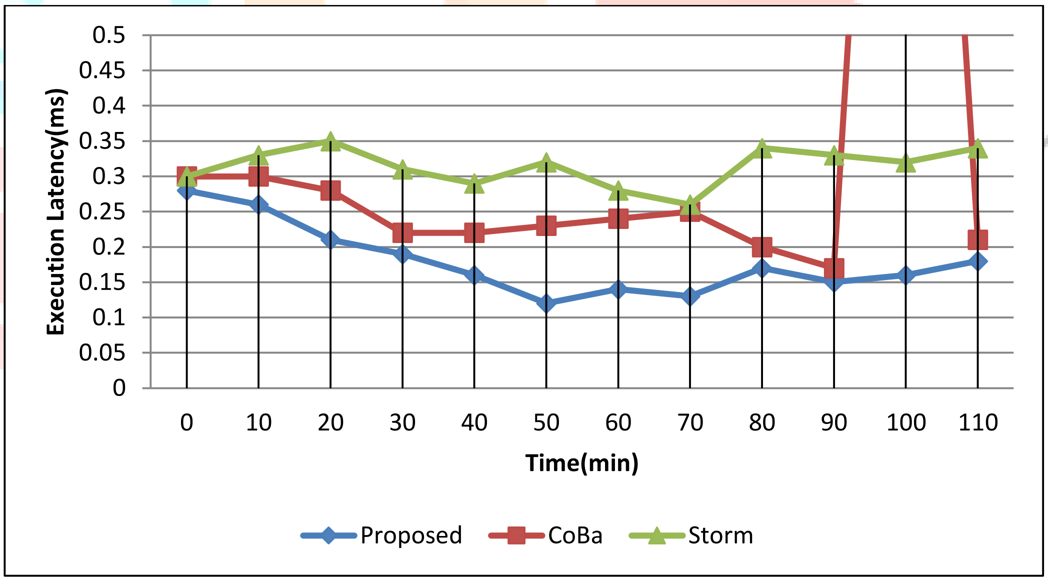
Profiling environment

The time consumed in creating schedules by different strategies (unit: milliseconds)

V. EXPERIMENTAL RESULTS

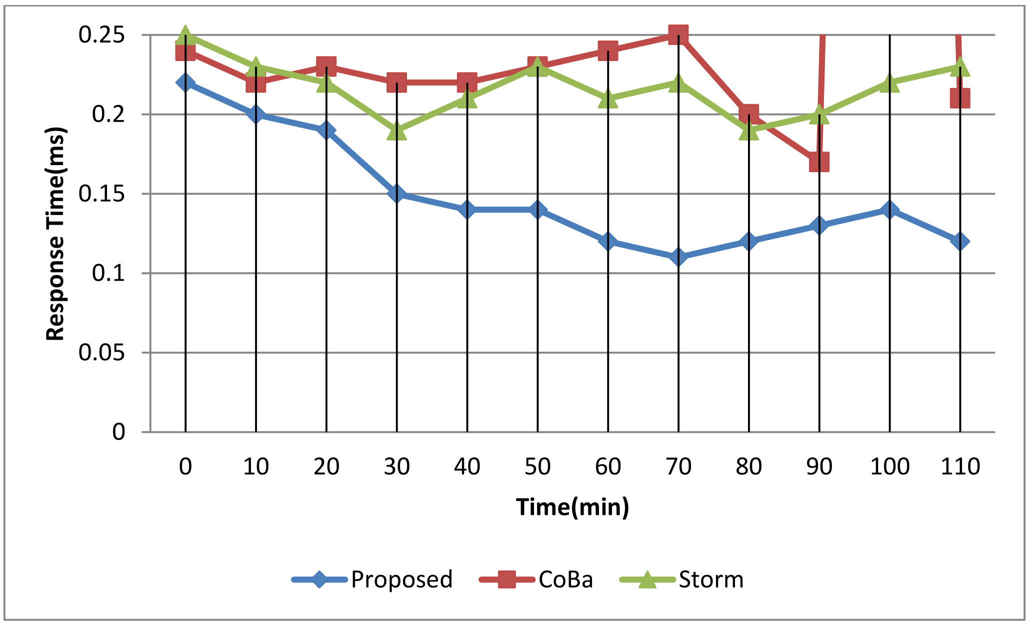
This section will show some simulation outcomes to evaluate the practicability of the devised optimization algorithm for data stream offloading and resource allocation. The metrics used for analyzing the system are Execution Latency, Response Time, Resource Utilization and availability.

The result in Figure 1.1 shows that the execution latency of proposed system is initially higher. This is because time is taken by the system to compute characteristics of big data stream before resources are allocated to it. Once calculated, appropriate resources are allocated by the system that results in lower execution latency for proposed system as compared with other tools.



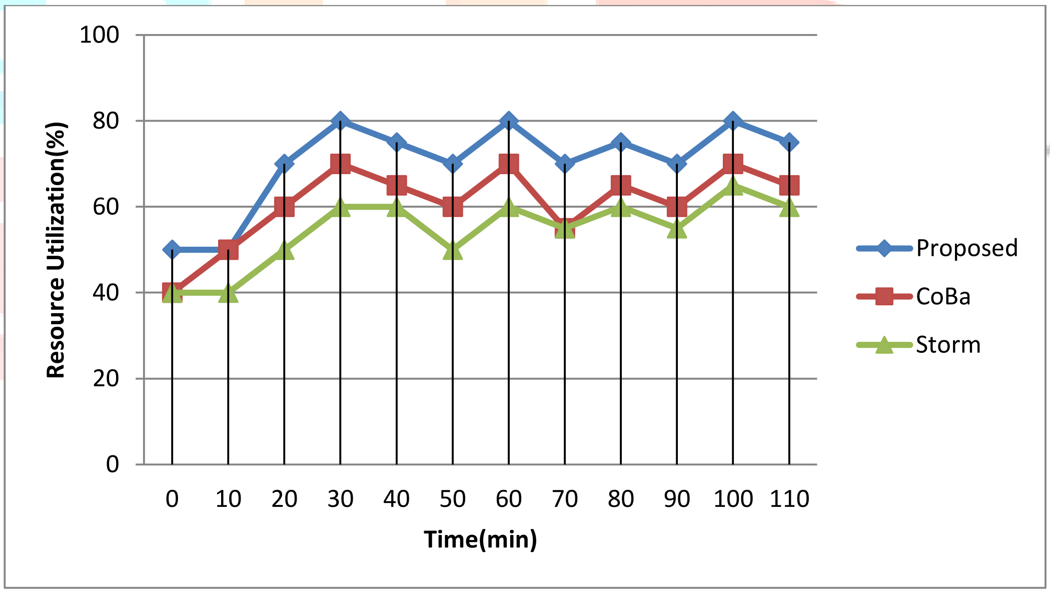
Result Figure1.1: Execution Latency Comparison

Figure 1.2 shows the response time of the system and other tools. It is observed that the response time roughly increases with the increase in time in case of CoBa and Storm. Response time for proposed system is less compare to other systems.



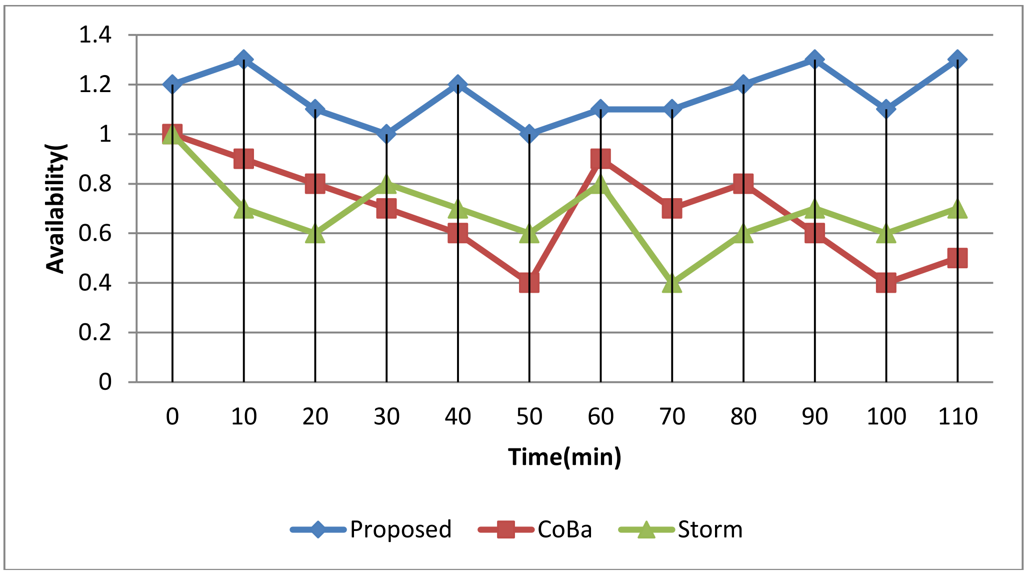
Result Figure1.2: Response Time Comparison

Figure 1.3 shows the comparison results of proposed system with other tools in resource utilization. The results show that utilization of cloud resources is higher in case of proposed system. This is due the fact that whenever characteristics of a stream changes, it is moved to a more suitable cluster according to data characteristics.



Result Figure1.3: Response Time Comparison

Figure 1.4 shows that resource availability is higher in case of proposed system as compared to other tools. The reason for higher resource availability is the topological ordered cluster formed by proposed system. The topological ordering allows big data stream to acquire other similar resources in case of nonavailability of required resources rather than waiting for busy resources.



Result Figure1.4: Availability Comparison

The results presented in this section suggest that the proposed system efficiently manages cloud resource in real time for big data streams.

VI. CONCLUSIONS AND FUTURE WORK

Distributed, large-scale processing of big data has a significant impact on both research and industry. To support user-specific SLA requirements and to maximize an resource cluster utilization, our research focuses on proposing a cost-effective resource allocation model. The aim is to allow the user a way of automatic and efficient deployments of applications in a local or cloud cluster. We have developed a profiler for which can be used to profile an application in the real cluster in terms of different resource allocation schemes and input workloads. Moreover, we have developed a light-weight resource allocation framework that can be plugged into the master node of the cluster. Applications can be submitted to progiler instead of directly submitting to the cluster. Based on the application profiles received from the profiler, uses the proposed resource allocation model to select a deadline-based costeffective resource allocation scheme to deploy an application to the cluster.

We have conducted experiments to evaluate the efficiency of our proposed models. In addition, we have shown the accuracy of the application completion time prediction model for three (3) different applications.

REFERENCES

[1]. Abbas, A., K. Bilal, L. Zhang, and S. U. Khan. 2015. “A Cloud Based Health Insurance Plan Recommendation System: A User Centered Approach.” Future Generation Computer Systems 43–44: 99–109.

[2]. Abolfazli, S., Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya. 2014. “Cloud-Based Augmentation for Mobile Devices: Motivation, Taxonomies, and Open Challenges.” IEEE Communications Surveys Tutorials 16 (1): 337–368.

[3]. Abouzeid, A., K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin. 2009. “HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads.” Proceedings of the VLDB Endowment 2 (1): 922–933.

[4]. Abraham, A., and M. Paprzycki. 2004. “Significance of Steganography on Data Security.” In Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC’04). Vol. 2, 347–351. IEEE.

[5]. Aghabozorgi, S., A. Seyed Shirkhorshidi, and T. Ying Wah. 2015. “Time-series Clustering – A Decade Review.” Information Systems 53 (C): 16–38.

[6]. Agrawal, D., P. Bernstein, E. Bertino, S. Davidson, U. Dayal, M. Franklin, J. Gehrke, et al. 2011. Challenges and Opportunities with Big Data 2011–1. Cyber Center Technical Reports. http://docs.lib.purdue.edu/cctech/1.

[7]. Agrawal, D., S. Das, and A. El Abbadi. 2011. "Big Data and Cloud Computing: Current State and Future Opportunities." In Proceedings of the 14th International Conference on Extending Database Technology, 530–533.

[8]. Aji, A., F. Wang, H. Vo, R. Lee, Q. Liu, X. Zhang, and J. Saltz. 2013. "Hadoop GIS: A High Performance Spatial Data Warehousing System Over Mapreduce." Proceedings of the VLDB Endowment 6: 1009–1020.

[9]. Alam, S., F. D. Albareti, C. A. Prieto, F. Anders, S. F. Anderson, B. H. Andrews, E. Armengaud, et al. 2015. "The Eleventh and Twelfth Data Releases of the Sloan Digital Sky Survey: Final Data from SDSS-III." The Astrophysical Journal Supplement Series 219 (1): 1–27.

[10]. Alvaro, P., T. Condie, N. Conway, K. Elmeleegy, J. M. Hellerstein, and R. Sears. 2010. "Boom Analytics: Exploring Data-centric, Declarative Programming for the Cloud." In Proceedings of the 5th European Conference on Computer Systems, 223–236. New York, NY: ACM.

[11]. Alyass, A., M. Turcotte, and D. Meyre. 2015. "From Big Data Analysis to Personalized Medicine for All: Challenges and Opportunities." BMC Medical Genomics 8: 1–33.

[12]. Aminzadeh, N., Z. Sanaei, and S. H. Ab Hamid. 2015. "Mobile Storage Augmentation in Mobile Cloud Computing: Taxonomy, Approaches, and Open Issues." Simulation Modelling Practice and Theory 50: 96–108.

[13]. Schroeck M, Shockley R, Smart J, Romero-Morales D, Tufano P. Analytics: the real-world use of big data. In IBM Global Business Services; 2012.

[14]. Gandomi A, Haider M. Beyond the hype: big data concepts, methods, and analytics. Int J Inf Manage. 2015;35:137-144.

[15]. TechAmerica Foundation: Federal Big Data Commission. A practical guide to transforming the business of government; 2012:1–40.

[16]. Cukier K. The economist, data, data everywhere: a special report on managing information, 2010, February 25; 2010. Retrieved from http://www.economist.com/node/15557443

[17]. Lazar N. The big picture: big data computing. Chance. 2013;26 (2):28-32.

[18]. Yang C, Huang Q, Li Z, Liu K, Hu F. Big data and cloud computing: innovation opportunities and challenges. Int J Digit Earth. 2016;8947:1-41.

[19]. Hashem IAT, Yaqoob I, Badrul Anuar N, Mokhtar S, Gani A, Ullah Khan S. The rise of 'big data' on cloud computing: review and open research issues. Inf Syst. 2015;47:98-115.

[20]. Su X, Gilman E, Wetz P, Riekki J, Zuo Y, Leppänen T. Stream Reasoning for the Internet of Things. In: Proceedings of the 6th International Conference on Web Intelligence, Mining and Semantics-WIMS '16. Nîmes, France: ACM; 2016:1-10.

[21]. Yasumoto K, Yamaguchi H, Shigeno H. Survey of real-time processing technologies of IoT data streams. J Inf Process. 2016;24(2):195-202.

[22]. Cortés R, Bonnaire X, Marin O, Sens P. Stream processing of healthcare sensor data: Studying user traces to identify challenges from a big data perspective. Procedia Comput Sci. 2015;52:1004-1009.

[23]. Sun D, Zhang G, Yang S, Zheng W, Khan SU, Li K. Re-stream: real-time and energy-efficient resource scheduling in big data stream computing environments. Inf Sci (Ny). 2015;319:95-112.

[24]. Tolosana-Calasanz R, Bañares JÁ, Pham C, Rana OF. Resource management for bursty streams on multi-tenancy cloud environments. Futur Gener Comput Syst. 2016;55:444-459.

[25]. Rahman M, Graham P. Responsive and efficient provisioning for multimedia applications. Comput Electr Eng. 2016;53:458-468.

[26]. Condie T , Mineiro P , Polyzotis N , Weimer M . Machine learning for big data. ACM SIGMOD international conference on management of data New York, USA; 2013.

[27]. Wu X , Zhu X , Gong-Qing W , Ding W . Data mining with big data. IEEE Trans Knowl Data Eng 2014;26(1):97–107 .

[28]. Gebara F , Hofstee H , Nowka K . Second-generation big data systems. IEEE Comput 2015;48(1):36–41 .

[29]. Abbas A , Adjeroh D , Dredze M , Paul MJ , Zahedi FM , Zhao H , et al. Social media analytics for smart health. Intell Syst IEEE 2014;2(29):60–80 .

[30]. Kejela G , Esteves RM , Rong C . Predictive analytics of sensor data using distributed machine learning techniques. IEEE 6th international conference on cloud computing technology and science (CloudCom); 2014 .

[31]. Feroz MN , Mengel S . Examination of data, rule generation and detection of phishing URLs using online logistic regression. IEEE international conference on big data; 2014.