IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-12, Dec 2018 E-ISSN: 2347-2693

Challenges and Opportunities of Resource-Aware Allocation Frameworks for Big data tools in Cloud Computing

R. Rengasamy[1*], M.Chidambaram[2]

[1]Department of Computer Science, Bharathidasan Government College for Women, Puducherry, India
[2]Department of Computer Science, Rajah Serfoji Government College, Thanjavur, India

[*]*Corresponding Author: rrsamy74@gmail.com , Tel.: +0091-431-7598883271*

Available online at: www.ijcseonline.org

Accepted: 22/Dec/2018, Published: 31/Dec/2018

***Abstract*-**System virtualization is the backbone of Cloud computing has been liberalizing its services to distributed data-intensive platforms such as MapReduce and Hadoop. Cloud computing empowers consumers to access online resources using the internet, from anywhere at any time without considering the underlying hardware, technical management and maintenance problems of the original resources. Cloud services are obtained from data centres which are distributed throughout the world. Big Data Applications with resource aware allocation has become an active research area in last three years. The Hadoop framework has been adopted to work efficiently in cloud computing. System virtualization is the backbone of Cloud computing, has been liberalizing its services to distributed data-intensive platforms such as MapReduce and Hadoop. Cloud computing empowers consumers to access online resources using the internet, from anywhere at any time without considering the underlying hardware, technical management and maintenance problems of the original resources.We present a detail study of various resource allocation and other scheduling challenges as well as frameworks for Hadoop Jobs in Cloud Computing.

***Keywords*-** Hadoop++, Cloud computing, MapReduce, YARN, Resource-allocation

I. INTRODUCTION

Cloud computing comprises of resources and services produced through the Internet. The buzzword Cloud computing is considered as a new computing paradigm which can provide customized, flexible, consistent, QoS guaranteed dynamic computing environments and its underlying capability to provide adjustable dynamic IT infrastructures and configurable software services. Grid computing has been replaced by cloud computing because of its system virtualization. Virtual resources are provided to clients through internet in cloud computing such as Gmail, provided by Google. The rapid growth in cloud computing has led to numerous advantages but at the same time it possesses lack of security concerns which has been a major challenge. There is a growing set of large-scale scientific applications which are loosely-coupled in nature comprises many small jobs/tasks with much shorter durations and encompass large volumes of data where these applications include those from data analytics, bioinformatics, data mining, astronomy, astrophysics, and MPI ensembles. Cloud computing is independent computing and varies from utility computing and grid computing. Technologies involved in Cloud computing are still developing and evolving for example, Service Oriented Computing. Cloud computing environment is still lacking security features and large-scale deployment and usage, which would rationalize the concept of Cloud computing and there exists no widely accepted definition [1]. Cloud computing has been expanding its services to data-intensive computing on distributed platforms such as MapReduce, Dryad, and Hadoop. In cloud-distributed platforms, virtualization takes place on the physical machines, and hence a virtual cluster is formed due to large collection of virtual machines. Data-intensive platform works on the virtual cluster unlike the traditional physical cluster. Such a virtual cluster provides adjustable environment, which can move up and down according to the changes in computation demands from different users. Cloud provider's combines virtual clusters from various users into a physical data centre, to increase the utilization of resources. In the virtual cluster, the need for computing resources for each node may change fluctuates, because of location of data and behaviour of task. However, a static cluster configuration is employed by current cloud services, manually adjusting the computing capability of each virtual machine deteriorate. The paper is organized as. Section 2 describes the important cloud metrics. Section 3 presents Hadoop MapReduce and its elastic resource management as well as challenges. Section 4 introduces elastic Resource Management in sustainable cloud datacentres. Section 5 presents various challenges and section 6 describes MapReduce framework modification and optimization. Section 7 presents conclusion.

II. CLOUD METRICS

In order to enhance cloud services, metrics must be considered since a metric delivers information about features of a cloud property by its parameters namely, unit, rules and

expression and the values emerging from the observation of the property. For example, to estimate a particular response time property from one client to another, we can consider the customer response time metric employed in a cloud email service search feature. By considering the customer response time metric, we can drive indispensable information which can employed for verification of observations as well as analysing the results. Metrics for cloud computing services can be described employing the Cloud Service Metric model which states higher level perceptions about the abstract metric definitions employed for a precise cloud service property which maybe a service uptime. Definitions for abstract metrics contain parameters and rules to direct a formal understanding the property of interest. The CSM model entails concrete metric definitions that are based on abstract metric definitions. Nevertheless, cloud metrology must be understood in a proper way. Frequent terminologies namely, definition of measurement, metrical groups of measurement artefacts, consists of have numerous definitions, these myriad terminologies become tedious for the cloud service customer to associate services or depend on third party tools to observe the health of the service. And cloud provider should be to analyse service whether it's operating correctly or to permit its service to arrive into an intricate cloud service federation. International organizations should construct a group of metrics which should be reliable, shareable and trustworthy. By defining the metrics; it not only increases the support of the decision-making process but enhances different phases of the cloud service lifecycle as shown in figure 1. Cloud computing is regarded as a competent data management which employs k-median clustering for managing data. Even though cloud computing entails high profitable technology for business value, it lacks certain security features. Hence it is obligatory to ensure that data and infrastructural resources in cloud are scheduled and data deduplication is carried out in an efficient way. Current frameworks have limitations and may not be resolving certain drawbacks encountered in cloud.

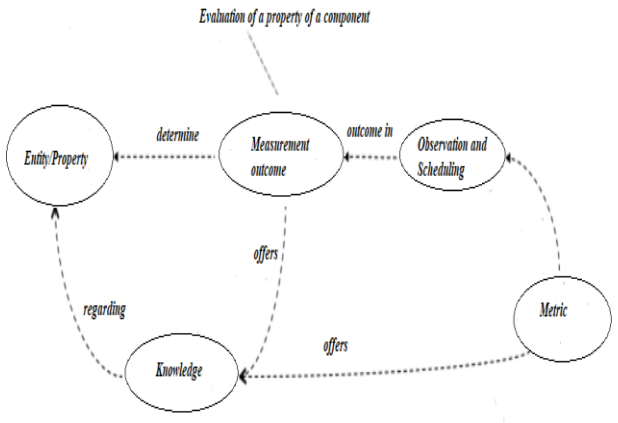


Figure 1: Cloud Metrics

The volume of data entering into cloud is rapidly increasing; therefore, on-demand cloud services should be provided to clients at any time, whereas providers must be able to preserve system availability as well as process a large volume of data. It is obligatory for cloud service providers to lessen large volumes of data; thereby they can reduce costs for maintaining large storage systems. Cloud storage employs data deduplication technique to enhance performance of cloud storage. The technique of data deduplication employed for enhancement of efficiency in storage. In a normal deduplication system, duplicated data recognize as well as collect a single copy of data in storage. Generations of logical pointers are employed for supplementary replicas as a replacement for storing redundant data. The main advantage of employing deduplication is that it reduces storage space and network bandwidth. The drawback of employing this technique is that it will take a toll on system tolerance since files denote to the identical chunk of data, if it turns out to be unobtainable due to failure which will in turn lessen reliability. Due to the drawback of static schemes, which cannot cope with changing user behaviour, deduplication in cloud storages requires a dynamic scheme which can adapt to various access patterns and changing user behaviour in cloud storages.

III. HADOOP MAPREDUCE: ELASTIC RESOURCE MANAGEMENT CHALLENGES

In the past few years, MapReduce has revolutionized parallel and distributed processing of Big Data. MapReduce is a parallel and distributed programming model on clusters of commodity hardware and has emerged as the de facto standard for processing a large set of unstructured data. It has proven to be an effective platform to implement complex batch applications as diverse as sifting through system logs, running extract-transform load operations, and computing web indexes. Since big data analytics requires distributed computing at scale, usually involving hundreds to thousands of machines, access to such facilities becomes a significant barrier to practising big data processing in small business. Deploying MapReduce in data centres or cloud platforms offers a more cost-effective model to implement big data analytics. As industries are confronting an unprecedented volume of data every day, Hadoop, the open source implementation of the MapReduce programming model, has become the *de facto* standard technique for storing and analysing peta scale data in a cost-efficient way. For example, the Data warehouse Hadoop cluster at Facebook contains 3000 machines and hosts on average 25000 MapReduce jobs per day [2]. However, study [3] has shown that current use of Hadoop in research and enterprises still has significant room for improvement on the performance of Hadoop jobs and the utilization of Hadoop clusters. There is a growing need for providing predictable services to Hadoop users who have strict requirements on job completion times (i.e., deadlines). However, meeting job deadlines is difficult

in current Hadoop platforms. First, because jobs have diverse resource demands, it is hard to determine how much resource is needed for each job to avoid its deadline miss. Second, Hadoop clusters are usually shared by multiple jobs and the scheduling order of these jobs can affect job completion time. Thus, allocating enough resources alone may not guarantee job completion time effectively. While existing schedulers in Hadoop, such as the default FIFO scheduler, Fair scheduler, Capacity scheduler, the RAS scheduler, and their variations, optimize job completion time without considering deadlines, there are recent studies that aim to guarantee job deadlines in Hadoop workloads by estimating job completion time and manipulating job queue ordering or task scheduling. A recent trend of running Hadoop in a hybrid environment further complicates the problem. To pursue cost-efficiency, Hadoop clusters can be powered by a mix of renewable energy and traditional power grid or share the same cloud infrastructure with interactive workloads or run opportunistically on transient resources, e.g., Amazon Spot Instances. In these scenarios, the resources available to the Hadoop cluster are quite dynamic due to the variable supply of renewable energy, the changing intensity of co-located workloads, or the abrupt termination of market-based resources. The dynamics in the capacity of Hadoop clusters pose significant challenges on satisfying job deadlines. First, it is hard to estimate job completion time with dynamically available resources. The prediction models should be robust to the varying cluster capacity. Second, job execution and task scheduling become more complicated. When the amount of available resources drops, high priority jobs or jobs with approaching deadlines should be prioritized to improve the application performance or revenue.

IV. ELASTIC RESOURCE MANAGEMENT IN SUSTAINABLE CLOUD DATACENTRES

To improve resource utilizations, data-centres consolidate workloads, ranging from transactional applications (e.g., e-Commerce website) to batch jobs (e.g., MapRedce data analytics), on the same physical hardware. These workloads are inherently heterogeneous with different QoS and resource requirements. A transactional workload comprises short client requests and its performance is measured based on the throughput of requests finished within a response time target. A MapReduce batch job is usually a long-running program with an expected completion time. While responsiveness is most important to transactional workloads as slow responses turn away potential customers, the delay in processing batch jobs is more tolerable and may be compensated later.

Another salient difference between the workloads is in their resource requirements. While the resource requirements of batch jobs is relatively stable during execution, the need of transactional workloads is quite dynamic [2][3] due to time-varying client traffic. Given a fixed total amount of resources, it is not trivial to determine the optimal resource allocations to these workloads. The dynamic power supply in a sustainable datacentre further complicates the problem. The varying power availability changes the amount of resources available to these workloads making an optimal allocation scheme both workload and power dependent. For an illustration, Figure 2 plots the resource requirements of representative transactional and batch workloads in various time intervals. We profiled the transactional resource usage from replaying the Wikipedia trace [4] and the batch resource usage from the Facebook analytic workload [5]. It illustrates that the batch workload volume is relatively stable during execution after the job is submitted. But for transactional workload, the workload volume (e.g., throughput) is quite dynamic. The unit on Y-axes is the dynamic workload volume of the transactional workload trace normalized to its beginning workload volume. The workload volume means throughput for RUBiS workload, and the size of input data for Hadoop workload.

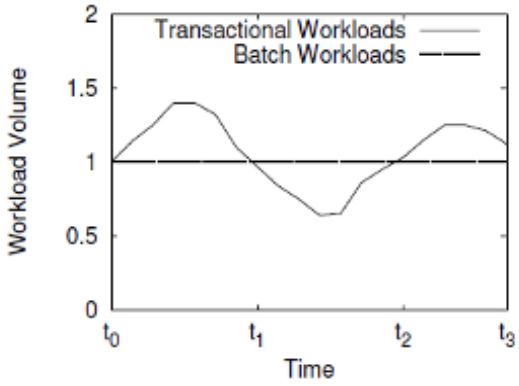
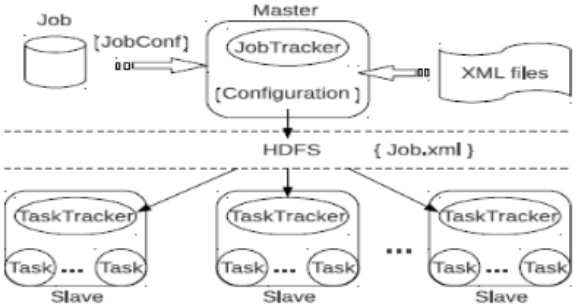


Figure 2 plots the resource requirements of representative transactional and batch workloads in various time intervals.

V. CHALLENGES

5.1: Inefficient Default Hadoop Configuration Policy

Based on the default Hadoop framework as shown in figure 3, many parameters need to be set before a job can run in the cluster. These parameters control the behaviours of jobs during execution, including their memory allocation, level of concurrency, I/O optimization, and the usage of network bandwidth.



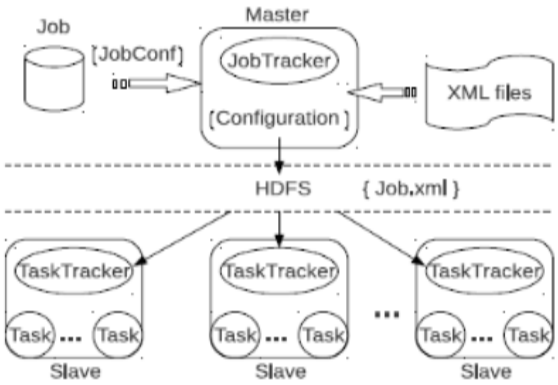


Figure 3: Architecture of Hadoop

In Hadoop, there are more than 190 configuration parameters, which determine the settings of the Hadoop cluster, describe a MapReduce job to the Hadoop framework, and optimize task execution [6]. Cluster level parameters specify the organization of a Hadoop cluster and some long-term static settings. Changes to such parameters require rebooting the cluster to take effect. Job-level parameters determine the overall execution settings, such as input format, number of map/reduce tasks, and failure handling. These parameters are relatively easier to tune and have uniform effect on all tasks even in a heterogeneous environment. Task-level parameters control the fine-grained task execution on individual nodes and can possibly be changed independently and on-the-fly at runtime. Hadoop installations pre-set the configuration parameters to default values assuming a reasonably sized cluster and typical MapReduce jobs. These parameters should be specifically tuned for a target cluster and individual jobs to achieve the best performance. However, there is very limited information on how the optimal settings can be determined. There exist rule of thumb recommendations from industry leaders (e.g. Cloudera and MapR) as well as academic studies. These approaches cannot be universally applied to a wide range of applications or heterogeneous environments.

5.2 Inefficient resource utilization in Spark-on-YARN

There exists a semantic gap between the reservation-based resource scheduling policy of YARN and the dynamic need of Spark applications, which causes inefficient resource utilization and poor application performance. Specifically, Spark-on-YARN raises several key challenges as follows: First, the reservation-based resource scheduling policy of YARN makes it hard for tasks with high resource demand to obtain the required resources in time. This has a big impact on Spark's performance. Due to Spark's multi-thread programming model, a single executor of Spark occupies a large amount of resource at one time. Thus, an executor with high resource demand may have to wait a long time for the resource reservation, leading low resource utilization and poor performance. Even worse, jobs with very large resource demand (e.g., Spark streaming) could suffer from starvation when the required resources cannot be obtained for a long period of time. Second, existing schedulers in YARN do not consider the impact of inter-task dependency between map and reduce phases. However, reduce tasks that are already launched cannot execute their functions until all map tasks are completed. As a result, the reduce tasks will keep occupying the resources without getting much work done. Thus, it incurs low utilization of the resources that are allocated to the reduce tasks.

5.3 Trade-off between Performance and Energy Efficiency

For most production Hadoop clusters, there are two main goals of operation: improving the performance of applications for increasing revenue and the energy efficiency of cluster system for reducing the operating cost. In addition, diverse resource demands from different jobs make the task of job scheduling more challenging, especially when the cluster is heterogeneous. Existing studies have shown that the performance of Hadoop jobs can be improved by various task scheduling solutions, e.g., Fair Scheduler, FLEX Scheduler and therefore, an efficient workload management strategy is critical for improving application performance. On the other hand, the operational expenditure on energy cost of the cluster is another significant concern for operators. Such architecture not only provides good system reliability, but also poses a large amount of reconfiguration cost due to necessary HDFS data replication requirement. Even in some cases, the idle nodes must remain powered on to ensure data availability. Therefore, a well deigned resource provisioning solution is challenging but important to achieve energy efficiency and improve system resource utilization.

VI.MAPREDUCE FRAMEWORK MODIFICATION AND OPTIMIZATION

Studies have demonstrated that it is effective to improve MapReduce performance by modifying the default Hadoop framework. There are growing interests on MapReduce performance optimization with various techniques, e.g., resource provisioning [8], job scheduling [10] and self-tuning configuration [11][8]. Rao *et al.* proposed Sailfish [12], a new MapReduce framework for large-scale data processing. The core of Sailfish is aggregating intermediate data, specifically data produced by map tasks and consumed later by reduce tasks, to improve job performance by batching disk I/O. Jinda *et al.* [13] proposed a new data layout, namely coined Trojan Layout, which internally organizes data blocks into attribute groups according to the workload in order to improve data access times. It can schedule incoming MapReduce jobs to data block replicas with the most suitable Trojan Layout. Guo *et al.* proposed iShuffle [14], a novel user-transparent shuffle service that provides optimized data shuffling to improve job performance. It decouples shuffle from reduce tasks and proactively pushes data to be shuffled to Hadoop node via a

novel shuffle-on- write operation in map tasks. Dittrich *et al.* proposed Hadoop++ [15], a new index and join technique to improve runtime of MapReduce jobs. They can schedule incoming MapReduce jobs to data block replicas with the most suitable Trojan Layout. Vavilapalli *et al.* proposed the next generation of Hadoop's compute platform–YARN [16], a new architecture introduced decouples the programming model from the resource management infrastructure.

6.1 Automated Parameter configuration for Hadoop

Recently, a few studies start to explore how to optimize Hadoop configurations to improve job performance. Herodotou *et al.* [17] proposed several automatic optimization-based approaches for MapReduce parameter configuration to improve job performance. Kambatla *et al.* [18] presented a Hadoop job provisioning approach by analysing and comparing resource consumption of applications. It aimed to maximize job performance while minimizing the incurred cost. Lama and Zhou designed AROMA [19], an approach that automated resource allocation and configuration of Hadoop parameters for achieving the performance goals while minimizing the incurred cost. AROMA achieves the optimal configuration by running a small sample of submitted jobs. If the workload is complex and dynamic, e.g., *Gridmix*, its profiling may not be accurate. Herodotou *et al.* proposed Starfish [20], an optimization framework that hierarchically optimizes from jobs to workflows by searching for good parameter configurations. It utilizes dynamic job profiling to capture the runtime behaviour of map and reduce at the granularity of phase level and helps users fine tune Hadoop job parameters. None of those approaches considered modifying the default Hadoop configurations to improve MapReduce performance. Verma *et al.* proposed a cluster resource allocation approach for Hadoop [21, 22]. They focused on improving the cluster efficiency by minimizing resource allocations to jobs while maintaining their service level objectives. They estimated the execution time of a job based on its resource allocation and input dataset, and determined the minimum resource allocation for the job. These approaches mostly rely on the default Hadoop framework and configure the parameters by static settings. They are often not effective when the workload changes or the cluster platform become heterogeneous.

6.2 Big data resource Management

YARN [23], the second generation of Hadoop, added a resource management layer in Hadoop. It allows different applications to be allocated with different number of task containers. Corona [24] is developed by Facebook and provides more flexibility to manage the cluster resources based on the different resource demands of workloads. Omega, a new parallel scheduler architecture built around shared state, using lock-free optimistic concurrency control, to achieve both implementation extensibility and performance scalability. Mesos abstracts CPU, memory, storage, and other compute resources away from machines, enabling fault-tolerant and elastic distributed systems to easily be built and run effectively. In these various Big-data resource managers, YARN is the only one for Spark that supports security and can leverage the existing HDFS dataset at the same time.

6.3 Resource Management for Multi-Tenant Hadoop Clusters

Sharing one Hadoop cluster among multiple users and jobs is the common practise for companies like Facebook and Yahoo [11][3]. But sharing a Hadoop cluster among multiple jobs from different users poses significant problems in resource allocation. There is a long history of work on resource management and scheduling in Hadoop [16][18]. There are a few studies that proposed different approaches to fairly allocate resources between multiple jobs. Quincy is a fair scheduler for Dryad that uses a centralized scheduling algorithm for Dryad's DAG-based programming model. It meets locality constraints of tasks by solving an optimization problem that includes the cost of migrating tasks. FLEX is a scheduling algorithm that enforces fairness between multiple jobs in a Hadoop cluster. It optimizes the performance of each job under different metrics. Upon guaranteeing fairness, FLEX is able to dedicate the unallocated resources to improve the performance of specific jobs. The Delay Fair Scheduler is an enhancement to Hadoop Fair Scheduler. It exploits the data locality of map task and significantly improves job performance [8]. It also supports multiple sub-clusters with the flexibility to apply different scheduling algorithms to them. The original design of Hadoop supports only limited control on resource allocation. There are recent new resource management and optimization frameworks for Hadoop as shown in figure 4.

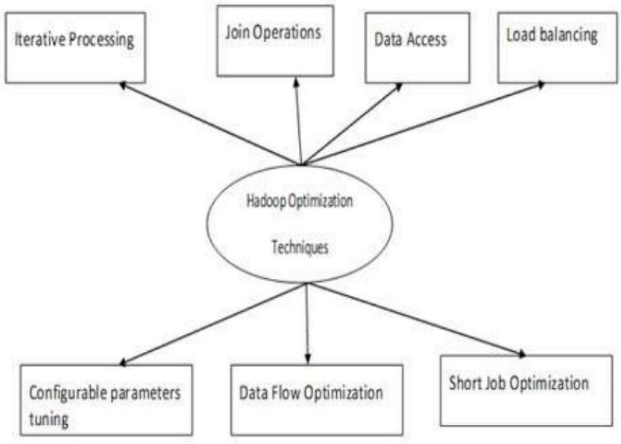


Figure 4: Hadoop Optimization Techniques

Mesos is a cluster manager that provides efficient resource isolation and sharing across distributed applications. Mesos simplifies the management of cluster by separating the

resource allocation and task scheduling. It only focuses on resource allocation, and delegates the task scheduling to a central pool of schedulers. YARN [24] is a resource management framework in Hadoop version 2. It has a two-level scheduling like Mesos, which separates the resource manager and the task scheduler. The available resources on each computation nodes are divided into containers. The resource allocation in YARN is performed upon the request from application. The task scheduling in YARN is delegated to the application. The new designs provide more flexibility in resource allocation and task scheduling for Hadoop. However, those resource allocation approach do not have fine-grained control on CPU resource. It takes the advantage of elastic resource allocation of the virtualized environment and provides a finer view and control of CPU resource as shown in figure 5.

Figure 5: Difference Schedulers for big data with hadoop

Scheduler	Job allocation	Environment		Priority in job queue	Resources sharing	Features	Drawbacks
		Homogeneous	Heterogeneous				
FIFO	Static	✓	✗	No	No	Easy to implement efficient	Data locality job starvation
Fair	Static	✓	✗	Yes	Yes	Fast response time mixing small with large job possible	Configuration problems unbalanced performance
Capacity	Static	✓	✗	No	Yes	Potential to reuse unused jobs in the queue	Complexity choosing scheduler
Delay	Static	✓	✗	Yes	No	Simple No overhead	No effective. Slots are limited
Match-making	Static	✓	✗	Yes	Yes	High data locality. Utilization level is high	--
LATE	Static	✓	✓	Yes	Yes	Heterogeneity more robust	Reliability
Deadline constraint	Dynamic	✓	✓	Yes	Yes	Supports optimization	Node must uniform
Resource aware	Dynamic	✓	✓	Yes	Yes	Performance resource utilization	Monitoring bottlenecks
Energy aware	Dynamic	✓	✓	Yes	Yes	Energy optimized	Multiple MapReduce jobs

6.4 Energy efficient Big Data Processing

There are growing interests on energy-efficient MapReduce design with various techniques, e.g., cluster consolidating [18], delaying batch jobs and load distribution. Leverich *et al.* proposed covering subset scheme keeps one replica of every block within a small subset of machines called the covering subset [25]. This subset remains fully powered to preserve data availability while the rest is powered down. It dynamically controls the available nodes in the cluster to improve the system resource utilization. To power down the whole cluster, AI strategy must run incoming jobs in regular batches, which would delay all jobs. Its key insight is that the interactive jobs can be served by a small pool of dedicated machines with their associated storage, while the less time-sensitive jobs can run in a batch fashion on the rest of the cluster. Those studies achieve energy efficiency by intrusive solutions, i.e., modify underlying HDFS file systems.

6.5 Sustainable Computing and Operation in Clouds

Datacentres have become a ubiquitous element of modern IT infrastructure, especially for the Internet services of cloud-based computing models. Global-scale online services typically run on hundreds of thousands of servers spread across dozens of datacentres worldwide, e.g., Google, Apple, Microsoft. These scales are increasing significantly as Infrastructure, Platform, and Storage-as-a-Service (IaaS, PaaS, and SaaS) models are more and more popular. These growing datacentres require considerable amount of electricity and are proliferating worldwide as a result of increased demand for IT applications and services. A recent study found that energy usage at datacentres is experiencing successive doubling every five years, and that the annual electricity cost at these centres can amount to $41.4 billion by 2015, which could make datacentres among the biggest greenhouse gas emitters by 2020. This huge IT energy consumption not only increases the total cost of operator but also leaves profound impact on the environment. As a result, concerns about the growth in energy usage and carbon emissions of datacentres have led to social interest in curbing their energy usage and emissions across their lifecycles. Sustainable computing research is hence becoming an important problem with the foremost of implications for future energy consumption and environmental impact.

VII. CONCLUSION

Big Data analytics and Cloud Computing have become two increasingly important IT paradigms to enhance business agility and productivity. As Cloud Computing offers a cost-effective way to support Big Data analytics, it exhibits many key characteristics. First, the user-perceived resource quantity and quality can vary significantly over time due to various interferences in the Cloud. Second, different types of Big Data applications, such as MapReduce and Spark, require distinct resource management schemes in the Cloud. Finally, sustainability is another concern for Cloud operators when consumers and organizations aggressively adopt cloud-based computing models.

REFERENCES

[1] Z. Abbasi, M. Pore, and S. K. S. Gupta. "*Online server and workload management for joint optimization of electricity cost and carbon footprint across data centers*". In *Proc. IEEE IPDPS*, 2014.

[2] R.Udendhran, "*A Hybrid Approach to Enhance Data Security in Cloud Storage* ", ICC '17 Proceedings of the Second International Conference on Internet of things and Cloud Computing at Cambridge University, United Kingdom — March 22 - 23, 2017, ACM ISBN: 978-1-4503-4774-7 doi>10.1145/3018.

[3] R Udendhran, K Muth Uramlingam, "*A dynamic data-aware scheduling for map reduce in cloud* ", Advanced Computing and Communication Systems (ICACCS), 2017 4th IEEE International Conference, **DOI:** 10.1109/ICACCS.2017.8014617.

[4] G. Ananthanarayanan, C. Douglas, R. Ramakrishnan, S. Rao, and I. Stoica. "*True elasticity in multi-tenant data-intensive compute clusters*". In Proc. of the ACM Symposium on Cloud Computing (SOCC), 2012.

[5] R. Appuswamy, C. Gkantsidis, D. Narayanan, O. Hodson, and A. Rowstron. "*Scale-up vs scale-out for hadoop: Time to rethink*" In Proc. ACM Symposium on Cloud Computing (SoCC), 2013.

[6] D. Borthakur, J. Gray, J. S. Sarma, K. Muthukkaruppan, N. Spiegelberg, H. Kuang, K. Ranganathan, D. Molkov, A. Menon, S. Rash, R. Schmidt, and A. Aiyer. "*Apache hadoop goes realtime at facebook*". In Proc. of the ACM SIGMOD, 2011.

[7] D. Carrera, M. Steinder, I. Whalley, J. Torres, and E. Ayguad,"*e. Enabling resource sharing between transactional and batch workloads using dynamic application placement*". In Proc. ACM/IFIP/USENIX Int'l Conf. on Middleware (Middleware), 2008.

[8] D. Carrera, M. Steinder, I. Whalley, J. Torres, and E. Ayguade. "*Autonomic placement of mixed batch and transactional workloads*". IEEE Trans. on Parallel and Distributed Systems (TPDS), 2012.

[9] R. Chaiken, B. Jenkins, P.-A. Larson, B. Ramsey, D. Shakib, S. Weaver, and J. Zhou. "*SCOPE: Easy and efficient parallel processing of massive data sets*". Proc. VLDB Endowment, 1(2):1265–1276, Aug. 2008.

[10] H. Chen, M. K. Cheng, and Y. Kuo." *Assigning real-time tasks to heterogeneous processors by applying ant colony optimization*". Journal of Parallel and Distributed Computing, 71, 2011.

[11] Y. Chen, S. Alspaugh, D. Borthakur, and R. Katz. "*Energy efficiency for large-scale mapreduce workloads with significant interactive analysis*". In Proc. of the EuroSys Conference (EuroSys), 2012.

[12] S. Rao, R. Ramakrishnan, A. Silberstein, M. Ovsiannikov, and D. Reeves. Sailfish*: "A framework for large scale data processing*". In Proc. of ACM Symposium on Cloud Computing (SoCC), 2012.

[13] A. Jinda, J. Quian-Ruiz, and J. Dittrich. "*Trojan data layouts: Right shoes for a running elephant*". In Proc. of ACM Symposium on Cloud Computing (SoCC), 2011.

[14] Y. Guo, J. Rao, and X. Zhou. "*shuffle: Improving hadoop performance with shuffle-on-write*". In Proc. Int'l Conference on Autonomic Computing (ICAC), 2013.

[15] J. Dittrich, J.-A. Quian´e-Ruiz, A. Jindal, Y. Kargin, V. Setty, and J. Schad." *Hadoop++: making a yellow elephant run like a cheetah (without it even noticing)*". In Proc. Int'l Conf. on Very Large Data Bases (VLDB), 2010.

[16] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O'Malley, S. Radia, B. Reed, and E. Baldeschwieler. "*Apache hadoop yarn: Yet another resource negotiator*". In Proc. ACM Symposium on Cloud Computing (SoCC), 2013.

[17] *H. Herodotou and S. Babu. "Profiling, what-if analysis, and cost-based optimization of mapreduce programs".* In Proc. Int' Conf. on Very Large Data Bases (VLDB), 2011.

[18] K. Kambatla, A. Pathak, and H. Pucha. *"Towards optimizing hadoop provisioning in the cloud". In Proc.* USENIX, HotCloud Workshop, 2009.

[19] P. Lama and X. Zhou." *Aroma: Automated resource allocation and configuration of mapreduce environment in the cloud*". In Proc. Int'l Conf. on Autonomic computing (ICAC), 2012.

[20] H. Herodotou, H. Lim, G. Luo, N. Borisov, L. Dong, F. B. Cetin, and S. Babu. "*Starfish: A self-tuning system for big data analytics*". In Proc. Conference on Innovative Data Systems Research (CIDR), 2011.

[21] A. Verma, L. Cherkasova, and R. H. Campbell. "*ARIA: automatic resource inference and allocation for mapreduce environments*". In Proc. of the ACM Int'l Conference on Autonomic Computing (ICAC), 2011.

[22] A. Verma, L. Cherkasova, and R. H. Campbell. "*Resource provisioning framework for mapreduce jobs with performance goals*". In Proc. ACM/IFIP/USENIX Int'l Middleware Conference (Middleware), 2011.

[23] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O'Malley, S. Radia, B. Reed, and E. Baldeschwieler. "*Apache hadoop yarn: Yet another resource negotiator*". In Proc. ACM Symposium on Cloud Computing (SoCC), 2013.

[24] Facebook. Hadoop corona: the next version of mapreduce. https://github.com/facebookarchive/hadoop-20/tree/master/src/contrib/corona.

[25] J. Leverich and C. Kozyrakis. "*On the energy (in)efficiency of hadoop clusters*". In Proc. USENIX HotPower, 2009.

Author Profile

R.Rengasamy is working as Assistant Professor in Bharathidasan Government College for Women (Autonomous), Puducherry. He received his PG Degree and M.Phil Degree from Bharathidasan University, Tiruchirapalli. He is currently pursuing Ph.D and he has 13 years of teaching experience. His Area of Research includes Cloud Computing and Big Data.

Dr. M. Chidambaram is working as Assistant Professor in Rajah Serfoji Government College (Autonomous), Thanjavur. He received his Ph.D Degree from Vinayaga Mission University, Salem. He has published more than 50 research papers in reputed national and international journals. His Area of Research includes Grid Computing, Cloud Computing and Big Data. He has 21 years of teaching experience and 5 years of Research Experience.

