Journal of The Gujarat Research Society

ISSN: 0374-8588
Volume 22 Issue 1, January 2020

# Using Network-Aware Task Assignment to Boost Map Reduce

Mr. Rajesh Pandey, Rajiv kumar
Shobhit Institute of Engineering and Technology (Deemed to be University), Meerut
Email Id- rajesh@shobhituniversity.ac.in, Rajiv.kumar@shobhituniversity.ac.in

***ABSTRACT:*** ***Running Map Reduce in a shared cluster to handle large-scale data analytical applications while increasing cluster utilization has been a current trend. However, network sharing across different apps may limit and heterogeneously distribute network capacity for Map Reduce workloads. As a result, network hotspots in racks become even more severe, rendering current task assignment rules that prioritize data location ineffective. This article provides a model to evaluate the connection between job completion time and the assignment of both map and reduce jobs across racks to address this problem. We also devise a network-aware task assignment method to reduce Map Reduce job completion times in shared clusters. It combines two basic but efficient greedy heuristics to decrease the time it takes to complete the map and reduce phases, respectively. We show that, when compared to state-of-the-art task assignment methods, the network-aware approach may reduce the average completion time of MapReduce tasks while maintaining an acceptable computing cost using large-scale simulations powered by Face book job traces.***

***KEYWORDS:*** ***MapReduce, Task assignment, Network hotspots, Software as a Service (SaaS).***

## 1. INTRODUCTION:

By processing enormous quantities of data in large-scale clusters, MapReduce has been extensively accepted as the fundamental method for powering commercial services in major IT firms like Google, Face book, and Yahoo. A current trend is to migrate MapReduce workloads from specialized clusters to shared clusters, such as Amazon EC2 and Mesos, in order to increase cluster usage. The speed of MapReduce applications may be substantially affected by network sharing in a shared cluster, which is a major issue. The bandwidth available for MapReduce applications becomes limited and diverse when the network resource is shared across virtual machines running multiple applications or among different computing frameworks. The study was funded in part by a grant from China's National Natural Science Foundation (NSFC) under project number 61133006.Three map (m1, m2, m3) and two reduce (r1, r2) jobs from a reduction-heavy project are assigned to three racks. The three input data blocks for map tasks areb1, b2, and b3. Dashed lines indicate congested connections and racks with limited network bandwidth.racks Furthermore, during the shuffle phase, reduce jobs must obtain data from all map tasks across racks through the network, thus network speed is critical to MapReduce performance[1]. As a result, network hotspots in shared clusters are more severe than in dedicated clusters, which will ultimately impact MapReduce task completion times. Existing work focuses on establishing data localization of map jobs or allocating each reduce task to the racks housing the most amount of its input data to relieve network hotspots and decrease the completion time for MapReduce applications. These systems, on the other hand, are intended for MapReduce to operate in a specialized cluster with ample of and homogenous network bandwidth in racks, and they optimize the map and reduce job assignments independently. With limited and diverse network bandwidth of racks accessible for MapReduce applications, there is a lack of attention paid to the assignment of both map and reduce jobs in a shared cluster.

As a consequence, even though map task assignment provides 100% data locality and zero cross-rack traffic, huge quantities of intermediate data produced by map jobs may congest racks with limited bandwidth resources during the shuffle phase. As an example, attaining data locality of map jobs greedily is not a guarantee of excellent performance of MapReduce applications on a shared cluster, as shown in Fig. 1. Because the bandwidth available for MapReduce on the rack hosting map jobs (Rack 1) is limited, huge quantities of intermediate data must be shuffled to reduce tasks, which take a long time. A network-aware approach, on the other hand, may relieve the network hotspot in Rack 1 by allocating two map tasks (m2, m3) to Racks 2 and 3. We present a network-aware job assignment method in shared clusters in this article. We get insight into the time bonus and penalty suffered by assigning a task to racks with diverse bandwidth by studying the connection between the assignments of both map and reduces tasks across racks and the completion time of MapReduce jobs. Through Boosting MapReduce with Network-Aware Task Assignment 81 the breakdown of the task assignment issue, we create our approach that combines two basic but effective greedy heuristics to reduce job completion time. We run lengthy simulations by replaying two MapReduce task traces from Facebook Inc. to assess the efficacy and overhead of our approach. In contrast to three previously suggested task assignment methods, we show that our networkaware approach can provide a speedup of 46.1–128.6% on average for MapReduce tasks. The remainder of this work is structured in the following manner. A model of MapReduce task completion time is presented in Section 2. The network-aware job assignment method is designed in Section 3. Section 4 assesses the strategy's efficacy and overhead. The related work is discussed in Section 5. Finally, in Sect. 6, we wrap up this study. 2 Modeling the Relationship between Job Completion Time and Task Assignment We initially simulate the connection between work completion time and task assignment in racks in this part. Then, in order to decrease work completion time, we construct an assignment issue that includes both map and reduce tasks. Figure 1 discloses An example of Algorithm 1: minimize the makespan T of map or reduce phase, when assigning 3 map or reduce tasks to 2 racks. The dashed rectangle means a time bonus.[2]

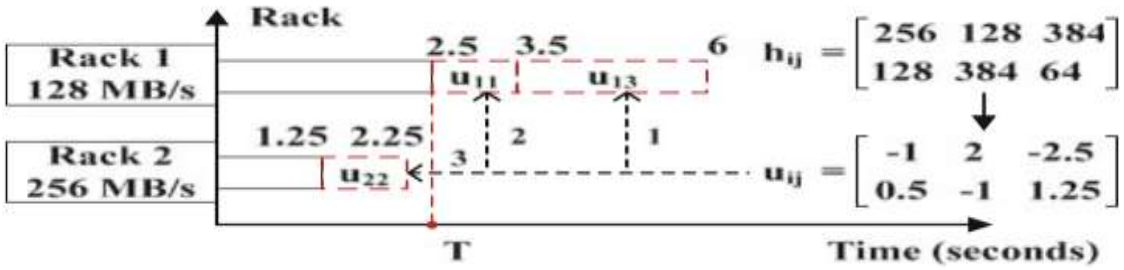


**Figure 1: An example of Algorithm 1: minimize the makespan T of map or reduce phase, when assigning 3 map or reduce tasks to 2 racks. The dashed rectangle means a time bonus.**

The customer is given the opportunity to deploy consumer-made or purchased applications built using programming languages, libraries, services, and tools supported by the provider into the cloud infrastructure. The customer has no control over the underlying cloud infrastructure, such as the network, servers, operating systems, or storage, but does have control over the installed apps and potentially the application-hosting environment's

configuration settings. PaaS providers provide application developers with a development environment. Typically, the supplier creates a toolset and standards for development, as well as distribution and payment methods. Cloud providers provide a computing platform in PaaS models, which usually includes an operating system, programming-language execution environment, database, and web server. Instead of purchasing and maintaining the underlying hardware and software layers, application developers build and execute their applications on a cloud platform. Some PaaS services automatically scale underlying computer and storage resources to meet application demand, removing the need for the cloud user to assign resources manually. [Some data integration and management companies also utilize PaaS-based customized apps as data delivery methods. iPaaS (Integration Platform as a Service) and dPaaS (Data Platform as a Service) are two examples (Data Platform as a Service). Customers may use iPaaS to create, execute, and manage integration flows. Customers control the creation and deployment of integrations using the iPaaS integration paradigm, which eliminates the need to install or manage any hardware or middleware. dPaaS provides completely managed integration and data management solutions. The PaaS provider, not the client, controls the creation and execution of programs by creating data applications for the customer under the dPaaS paradigm. Data visualization tools are used by dPaaS users to retrieve data. The customer is given the opportunity to utilize the provider's apps that are hosted on a cloud architecture. The apps may be accessed through a thin client interface, such as a web browser (e.g., web-based email), or a program interface, via a variety of client devices. With the potential exception of restricted user-specific application configuration options, the customer does not manage or control the underlying cloud infrastructure, which includes network, servers, operating systems, storage, or even individual application capabilities. Users may access application software and databases via the software as a service (SaaS) paradigm.

The infrastructure and platforms that operate the apps are managed by cloud providers. SaaS is also known as "on-demand software," and it is often paid on a pay-per-use or subscription basis. Cloud providers install and run application software in the cloud, while cloud customers access the program via cloud clients under the SaaS model. The cloud infrastructure and platform on which the application operates are not managed by cloud users. This removes the requirement for the cloud user to install and operate the program on their own machines, making maintenance and support easier. Scalability is a feature of cloud applications that may be accomplished by cloning jobs onto numerous virtual machines at runtime to accommodate changing work demand. Load balancers spread the workload over a group of virtual computers. The cloud user sees just one access point, thus this procedure is clear to them. Cloud applications may be multitenant to support a high number of cloud users, meaning that a single server can service several cloud-user organizations. Because SaaS apps usually charge a fixed cost per user on a monthly or annual basis, rates become scalable and changeable when users are added or withdrawn at any time. It's also possible that it'll be free. Proponents argue that by outsourcing hardware and software maintenance and support to the cloud provider, a company may decrease IT operating expenses. This allows the company to reallocate IT operations expenditures away from hardware/software and human costs and toward other objectives. Furthermore, since programs are hosted centrally, updates may be sent without requiring users to install new software. One disadvantage of SaaS is that customers' data is stored on the cloud provider's server. As a consequence, there may be illegal access to the data[citation required]. Games and

productivity tools like Google Docs and Word Online are examples of SaaS apps. SaaS applications may be linked to cloud storage or file hosting services, as Google Docs is with Google Drive and Word Online with One Drive.

### *1.1. Article in its entirety: Backend as a service for mobile devices*

Web app and mobile app developers are given a way to link their applications to cloud storage and cloud computing services with application programming interfaces (APIs) exposed to their applications and custom software development kits in the mobile "backend" as a service (m) model, also known as backend as a service (BaaS) (SDKs). User administration, push alerts, integration with social networking sites, and other services are available. Most BaaS companies are from 2011 or later, making this a relatively new cloud computing paradigm. However, current trends show that these services are acquiring considerable mainstream momentum among corporate customers. Serverless computing, also known as Function-as-a-Service, is a kind of computing that does not need a server (FaaS)Serverless computing is a cloud computing code execution model in which the cloud provider manages the start and stop of virtual machines as needed to serve requests, and requests are billed by an abstract measure of the resources required to satisfy the request rather than per virtual machine, per hour. Despite the name, it does not include the execution of programs without the need of servers. The term "server less computing" comes from the fact that the owner of the system does not have to buy, rent, or supply servers or virtual machines for the back-end code to operate on. Function as a service (FaaS) is a server less computing-based remote procedure call that enables the deployment of specific functions in the cloud that execute in response to events. Some people consider FaaS to be a subset of server less computing, while others use the words interchangeably. Private cloud refers to cloud infrastructure that is administered exclusively for the benefit of a single company, whether it is managed internally or by a third party, and is hosted domestically or outside.

A private cloud project requires substantial involvement in order to virtualize the business environment, as well as a reevaluation of current resource choices. It may boost profits, but every stage of the process presents security concerns that must be addressed to avoid severe flaws. Self-contained data centers are often expensive to operate. They have a large physical footprint, necessitating space, hardware, and environmental controls allocations. These assets must be renewed on a regular basis, which necessitates extra capital expenditures. They've been chastised since customers "still have to purchase, construct, and maintain them" and so don't profit from less hands-on management, basically "[lacking] the economic paradigm that makes cloud computing such an attractive idea." See Cloud-computing comparison for a comparison of cloud-computing software and services. When cloud services are provided via the public Internet, they are called "public," and they may be available for a fee or for free. There are minimal architectural differences between public and private cloud services, but when services (applications, storage, and other resources) are shared by many users, security issues rise dramatically. The majority of public cloud providers provide direct-connection services, which enable clients to securely connect their traditional data centers to cloud-based applications. Several variables influence whether businesses and organizations select a public cloud or on-premises solution, including solution functionality, pricing, integrational and organizational elements, as well as safety and security. The term "hybrid cloud" refers to a combination of a public cloud and a private environment, such as a private cloud or on-premises resources, that stay separate but are linked to provide the advantages of various

deployment methods. The capacity to link collocation, managed, and/or dedicated services with cloud resources is referred to as hybrid cloud. A hybrid cloud service, according to Gartner, is a cloud computing solution that combines private, public, and community cloud services from several service providers. [1 A hybrid cloud service spans isolation and provider borders, making it impossible to categorize it as either private, public, or community cloud. It enables you to increase a cloud service's capacity or capabilities by aggregating, integrating, or customizing it with another cloud service.

Hybrid cloud composition has a wide range of applications. For example, a company could keep confidential client data on a private cloud application in-house, but link it to a business analytics application hosted on the cloud.

## 2. DISCUSSION:

The goal of the task assignment issue is to decrease work completion time by optimizing task assignment across racks for both map and reduce activities.max I (wm I m + cm I ) + maxi

The preceding formulation is a 0-1 integer min-max optimization issue that looks like an Imbalanced Time Minimizing Assignment Problem and has been shown to be NP-hard . Although a dynamic programming method might be used to tackle this issue, the slowness of iterating over all r(p+q) potential solutions would be a significant disadvantage. We want to develop a heuristic task assignment method that might be applied in a real-world cluster since our goal is to reduce job completion time without adding too much computing cost to MapReduce.A Task Assignment Strategy That Is Network-Aware In this part, we deconstruct's task assignment issue into two sub problems: first, minimizing the makespan of a single (map or reduce) phase, and then minimizing the makespans of the map and reduce phases together[3].

First, we'll tackle the single-phase problem: How can we allocate map or reduce jobs to racks to shorten the map or reduce phase's make pan given the (map- or reduction-) input data contained in racks The initial makespan of the map or reduction phase is jK cij. A time bonus or penalty uij is also applied to Ti in Ri when a map or a reduce job j is assigned to Ri. As a result, Algorithm 1 is designed to greedily reduce the maximum of Ti by allocating the job with the lowest uij to each rack one by one until all tasks are allocated.1 It's worth noting that we start the following wave once we've filled all of the available spaces in the racks. the task computation time (wm I m, wr I r) is fixed as p/ iR sm I m, q/ iR sr I r. For the sake of simplicity, it is ignored while computing the phase makespan. When allocating three map or reduce jobs to two racks, an example of Algorithm 1 is to minimize the makespan T of the map or reduce phase. A time bonus is shown by the dashed rectangle. Consider the illustration in Figure 2. Algorithm 1 allocates a starting time to each rack, such as T1 = jK h1j 128 = 6, T2 = jK h2j 256 = 2.25, and uij is produced by Eq. Because uij contains three components that are all smaller than zero, we choose R1 with the longest execution time Ti and give job 3 with the shortest uij to it. T1 becomes 62.5=3.5 as a consequence. After that, we give task 1 to R1 and task 2 to R2 in that order. The map or reduction phase ultimately has a makespan of 2.5.Algorithm 1: Minimize the map's makespan or decrease the phase As long as there are unassigned jobs, 3: If the set of racks R that satisfy uij 0 is empty, 4: Find a rack I R with the lowest Ti 5: Otherwise Locate a rack I R with the highest Ti 7: End if 8:

Assign the assignment j K to the rack I with the lowest uij. the task assignment set X is returned, whereas 11: the task assignment set X is ended. Observation The process of Algorithm 1 corresponds to the Map Reduce job scheduling system, in which the master node distributes tasks one by one based on heartbeats from each slave node (rack)[4] . Furthermore, Algorithm 1 has a complexity of After that, we solve the two stages issue, which is a multiobjective optimization problem: How can we allocate map tasks to racks, given the map-input data hm ij and reduce-input data produced by map tasks Lj, to decrease the map phase and first reduce phase make spans together We use the weight sum technique to find a Pareto optimum solution for such an optimization issue, which minimizes sum2 i=1iTi(X), i.e., minimizes the sum of map phase makespan and initial reduce phase makespan when The weights I are set to 1 in Boosting Map Reduce with Network-Aware Task Assignment 85. As a result, Algorithm 2 is designed to greedily minimize the total of increased map phase and reduce phase make spans by allocating the map task with the largest reduction-input data Lj to racks one at a time , until all map tasks are allocated. The enlarged make spans of the map phase and reduction phase, respectively, are T m and Tr. Consider the illustration in Figure 3. Three map jobs' input data blocks (m1, m2, and m3) are stored in R1, R2, and R1, respectively. Because each map job j generates 384, 128, and 512 MB of reduce-input data Lj, Algorithm 2 selects to allocate m3 with the most reduce-input data to racks first. T m = 0 since the map-input data of m3 is kept in R1, and Tr = 512 128 = 4 if m3 is allocated to R1. T m = 128 256 = 0.5 if m3 is allocated to R2, since m3's map-input data is transmitted to R2, and TR = 512 256 = 2. As a result, allocating m3 to R2 results in the smallest sum of T m and Tr. To minimize the sum of T mand Tr, we allocate m1 to R1, m2 to R2 in the same way as we assigned m3 to R3.

When allocating three map jobs to two racks, Algorithm 2 is used to jointly minimize themakespan of the map phase T m and the starting makespan of the reduction phase Tr. Algorithm 2: Minimize the map phases and reduction phase's initial make spans together. Set up T m I Tr I 0, I R; a set of racks with available slots Redo while there are unassigned map tasks Find a map assignment with the greatest Lj of j Km..finish, while 8: return the Xm map task assignment Observation Algorithm 2 works with Algorithm 1 to reduce the time it takes for MapReduce tasks to complete. After Algorithm 2 produces a map task assignment set Xm that reduces the makespan of the map phase and the starting makespan of the reduce phase, Algorithm 1 finds a reduce task assignment set Xr that reduces the makespan of the reduce phase even more. Furthermore, Algorithm 2 has a complexity of O(p r).In this part, we assess our strategy's efficacy and computing overhead. We created a simulator with 1,100 lines of C code that was powered by two one-day MapReduce task traces from Face book Inc..

A 600-server shared cluster was set up[5]. The settings are selected in accordance with Microsoft's Cosmos cluster to make the simulation realistic. The number of servers in a rack is set to 30, the number of racks r is set to 20, the number of map, reduce, and sm, sr slots in a rack is set to 60, the size of a data block is set to 128 MB,[6] and the network bandwidth capacity of a rack is set to 10 Gbps. In addition, we randomly create background traffic in each rack and set the average computation time of a map job m and a reduction task r to 5 s and 3mp q s, respectively.1 Network-Aware Task Assignment's Effectiveness To demonstrate the efficacy of our network-aware task assignment approach, we execute all, 043 Map Reduce tasks in traces and compare it to three widely-used strategies: random assignment, original Time to Finish the Job (seconds)Random Orig. Map Reduce CDF percent of jobs Task completion time with LARTS Network Aware (b) job completion time

with LARTS Network Aware (c) job completion time with Random Orig. MapReduce CDF[7] percent of jobs cross-rack traffic using LARTS NetworkAwareLocality of Map DataRandom Orig. Map Reduce CDF percent of jobs ARTS Network Aware map task data locality Random: Map and reduce jobs are allocated to available racks in a random order. Original Map Reduce Each map job achieves data locality in a greedy manner. Reduce jobs are allocated to available racks at random. Each map job accomplishes data locality in a greedy manner. Each reduction job is given to the rack that [8]has the most input data. In comparison to the three previous task assignment methods, demonstrates that our approach may save a substantial amount of time to finish Map Reduce operations while incurring minimal cross-rack traffic and compromising the data locality of map tasks for approximately 50% of workloads. As demonstrated in, the network-aware approach beats the other techniques in terms of task completion time. The reason for this is because our approach considers the network bandwidth of racks and optimizes the assignment of map and reduces jobs to racks together. Boosting Map Reduce with Network-Aware Task Assignment 87, as shown in, while LARTS delivers the least cross-rack traffic of the other methods, it is not a guarantee of excellent Map Reduce performance. Because the chosen rack housing the maximum input data of a reduction job is a network hotspot, the network transfer of cross-rack traffic is unavoidably prolonged. Furthermore, with our approach, cross-rack traffic is somewhat higher than with LARTS and original Map Reduce. In comparison to the original Map Reduce and LARTS, demonstrates that our approach violates 100% map data locality. The reason is because, when assigning map tasks, our approach takes into account both the beginning makespan of the map phase and the initial makespan of the reduction phase of the job. The mean values of the simulation results are quantitatively shown in Figure 2 Assigning 3 Map (M1, M2, M3) and 2 Reduce Tasks (R1, R2) Of A Reduce-Heavy Job to 3 Racks. B1, B2, B3 Are the Three Input Data Blocks For Map Tasks. Congested Links And Racks With Constrained Network Bandwidth Are Labeled With Dashed Lines[9].

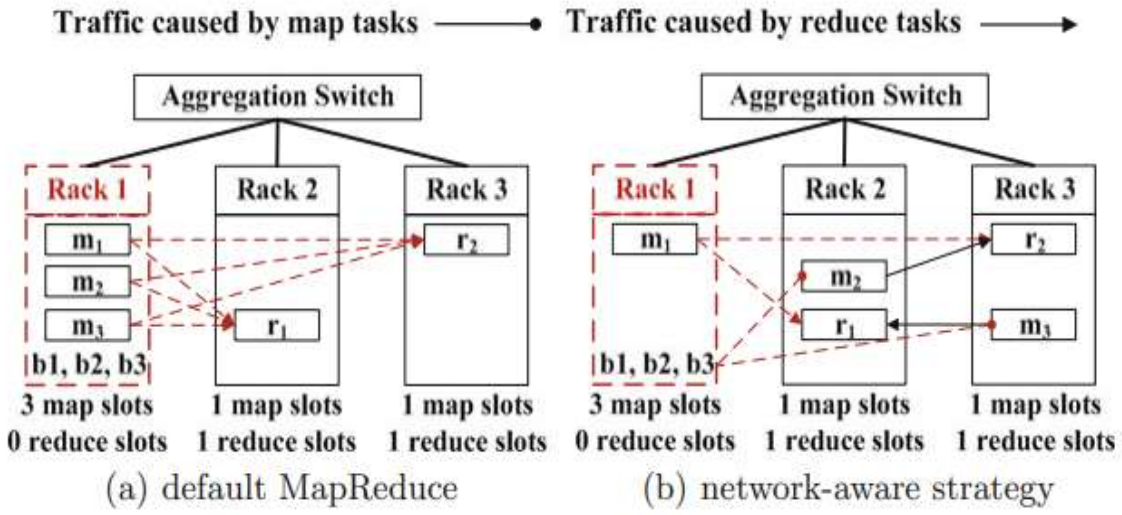


**Figure 2: Assigning 3 Maps (M1, M2, M3) and 2 Reduce Tasks (R1, R2) Of A Reduce-Heavy Job to 3 Racks. B1, B2, B3 Are the Three Input Data Blocks For Map Tasks. Congested Links And Racks With Constrained Network Bandwidth Are Labeled With Dashed Lines.**

Infrastructure as a service (IaaS) refers to online services that provide high-level APIs for abstracting different low-level aspects of underlying network infrastructure, such as physical computer resources, location, data partitioning, scalability, security, and backup. The virtual computers are operated as guests by a hypervisor. Large numbers of virtual machines may be supported by pools of hypervisors inside the cloud operating system, as well as the flexibility to scale services up and down based on client needs. Linux containers are separate partitions of a single Linux kernel that runs on real hardware.

The underlying Linux kernel technologies utilized to isolate, secure, and manage the containers are Linux cgroups and namespaces. Because there is no hypervisor overhead, containerization provides better performance than virtualization. Additional resources such as a virtual-machine disk-image library, raw block storage, file or object storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software packages are often available in IaaS clouds. Where the consumer is able to install and execute any software, which may include operating systems and applications," according to the NIST definition of cloud computing. The customer has no control over the core cloud infrastructure, but does have control over operating systems, storage, and installed applications, as well as potentially limited control over certain networking components (e.g., host firewalls)." These resources are available on demand from IaaS-cloud providers' vast pools of equipment in data centers. Customers may utilize the Internet or carrier clouds for wide-area connection (dedicated virtual private networks). Cloud customers install operating system images and application software on the cloud infrastructure to deploy their applications. The operating systems and application software are patched and maintained by the cloud user under this approach. IaaS services are usually billed on a utility computing basis by cloud providers: The quantity of resources provided and used is reflected in the cost.

## 3. CONCLUSION

This paper investigates the connection between job completion time and the assignment of both map and reduces tasks, with a special emphasis on the diverse bandwidth of racks, in order to minimize network hotspots and decrease the completion time of MapReduce processes in shared clusters. By simultaneously optimizing the assignment of both map and reduce tasks across racks, we further improve the network aware task assignment method to decrease job completion time. When compared to previously published task assignment methods, extensive simulation findings using real-world job traces show that our network-aware approach may reduce the average completion time of MapReduce jobs by 46.1–128.6%.

We want to implement our network-aware job assignment method in Hadoop and assess its efficacy in a shared cluster as part of our future study. We also want to include disk I/O in our model and expand our task assignment method, since this is another important aspect that affects MapReduce performance[10].

**REFERENCE**

[1] C. L. Rezende *et al.*, “From hotspot to hopespot: An opportunity for the Brazilian Atlantic Forest,” *Perspect. Ecol. Conserv.*, 2018.

[2] Z. Zhang, L. Yang, and Y. Zheng, “Translating and Segmenting Multimodal Medical Volumes with Cycle- and

Shape-Consistency Generative Adversarial Network," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.

[3] R. Mur-Artal and J. D. Tardós, "ORB-SLAM: Tracking and Mapping Recognizable," *Work. Multi VIew Geom. Robot. - RSS 2014*, 2014.

[4] Z. Ying, Z. Lu, and H. Dai, "Improving efficiency and patient satisfaction in a peripherally inserted central catheter center using lean-based methodology," *JAVA - J. Assoc. Vasc. Access*, 2014.

[5] A. Jain, T. Nueesch, C. Naegele, P. M. R. Lassus, and C. H. Onder, "Modeling and Control of a Hybrid Electric Vehicle with an Electrically Assisted Turbocharger," *IEEE Trans. Veh. Technol.*, 2016.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2015.

[7] G. I. Sayed, A. Darwish, and A. E. Hassanien, "A new chaotic multi-verse optimization algorithm for solving engineering optimization problems," *J. Exp. Theor. Artif. Intell.*, 2018.

[8] D. Karthika Renuka, P. Visalakshi, and S. P. Rajamohana, "An ensembled classifier for email spam classification in hadoop environment," *Appl. Math. Inf. Sci.*, 2017.

[9] D. Sánchez-González, L. M. A. Rivera, and V. Rodríguez-Rodríguez, "Natural landscape and healthy ageing in place: The case of the Cumbres de Monterrery National Park in Mexico," *Boletin de la Asociacion de Geografos Espanoles*. 2018.

[10] N. Molyneux, G. R. Da Cruz, R. L. Williams, R. Andersen, and N. C. Turner, "Climate change and population growth in timor leste: Implications for food security," *Ambio*. 2012.