Int. J. Advance Soft Compu. Appl, Vol. 8, No. 3, December 2016
ISSN 2074-8523

Performance Evaluation of Cloud Centers with High Degree of Virtualization to provide MapReduce as Service

C. N. Sahoo[1], Veena Goswami[1]

[1]School of Computer Applications
KIIT University, Bhubaneswar-751024, India

Abstract

MapReduce has emerged as a paradigm where massive amounts of data are parallel processed with the help of clusters. Hadoop, as an open source implementation, has been used in a variety of applications such as social networking, video and image processing, log analysis and search indexing etc. For enterprises, providing MapReduce as a service in the cloud becomes an attractive model. Cloud Systems providing MapReduce as a service will allow users to access large number of machines in a cost-effectively manner without creating the infrastructures of their own. Moving MapReduce to the virtualized environment will incur new challenges, because the computation model is strongly bound to data, its storage, and location which make its behavior rather a batch processing. We consider cloud centers where tasks arrive in batches or groups of random size and task service times are assumed to follow an exponential distribution. This paper also examines the cases where the arrival group size has a geometric distribution or a deterministic distribution. We examine a new analytical model for evaluation of performance of such large scale systems and compute the performance benchmarks such as mean waiting time in the queue, mean request response time, mean system length and the mean number of busy servers in the system.

Keywords: *Cloud Computing, MapReduce, Hadoop, Performance Evaluation, Virtualized data center, Batch processing.*

1 Introduction

Cloud computing offers a delivery model with virtually unlimited computing and storage capacity. Cloud is an attractive option for setting up and maintaining large-scale as well as complex infrastructure such as a Hadoop. It is really difficult to justify the bulk investment required in new infrastructure and continuous maintenance cost incurred while using a Hadoop cluster. Whereas Cloud allows rented model in a ”pay-per-use” manner which is really cost-effective and hassle free. So, instead of acquiring and maintaining own infrastructure, it is affordable and viable to host in cloud.

MapReduce [3] is a distributed programming platform designed for large scale computation over massive amounts of data. Hadoop, as an open source implementation is largely used in log analysis, image and video processing, data mining in social-networking sites and search indexing [1]. Henceforth, Hadoop and MapReduce will be used interchangeably in this article. Hadoop is characterized by fault tolerance, high efficiency and the ability to automatically parallelize applications on a cluster with thousands of hosts [3, 14]. Even with minimal distributed programming experiences, users can easily leverage a large cluster. MapReduce framework has formed the stack of technologies in empowering big enterprises such as Google, Facebook and Yahoo etc. A recent Gartner survey shows that 39% of enterprises have planning IT budgets for cloud computing [7]. So, providing MapReduce as a service in the cloud becomes an attractive usage model for enterprises [6]. MapReduce as cloud service will allow customers to cost-effectively rent a large number of machines in a cluster without creating the infrastructures of their own. They will be able to effectively size the MapReduce cluster according to their demand. Virtualized data center is the most common cloud computing platform. However, moving MapReduce to the virtualized environment will incur new challenges.

This paper focuses on an analytical model through which we obtain mean request response time, probability distribution of number of tasks, probability of immediate service and other important performance indicators in order to ensure the Quality of service(QoS). In order to obtain immediate service for a request, the cloud service provider can tune the number of servers to be deployed at the service end. To fulfill the service level agreements (SLA), it requires exact performance evaluation so that the service providers can decide upon the size of resources to be deployed.

The rest of this paper is described as follows. A brief description of the system model is narrated in Section 2 and its analysis for the Cloud computing environments hosting BigData applications. Analytical model and its analysis is given in Section 3. Section 4 depicts some important performance measures. Numerical analysis is done in Section 5 in order to show the efficiency of system parameters. Conclusion the paper is in Section 6.

2 System Model

The Hadoop programming model is based on the following simple concepts: (i) iteration over the input; (ii) computation of key/value pairs from each piece of input; (iii) grouping of all intermediate values by key; (iv) iteration over the resulting groups; (v) reduction of each group [3, 11]. Each job is partitioned into a number of map and reduce tasks. Each Mapper (map task) runs map functions on single data block (128MB / 64MB) and each data block is replicated 3-times. A Reducer (reduce task) generates the final results based on the intermediate results from the map tasks. There is a barrier synchronization between map and reduce phase. It means that all map tasks have to be completed prior to any reduce task can start, that is, the computation model is strongly bound to data, its storage, and location. This behavior makes MapReduce rather a batch processing than online tool. A master (Namenode) and multiple slaves (Datanodes) are there in a typical MapReduce program. Management of the framework, job queue organization, user interaction and task scheduling are different responsibilities of the Master. In order to perform tasks, a fixed number of map and reduce slots are provided to each slave. Through a heartbeat protocol, each slave reports regarding the number of free task slots available to the job scheduler which is located in the master. MapReuce data flow with multiple reduce tasks is represented in the Figure 1.

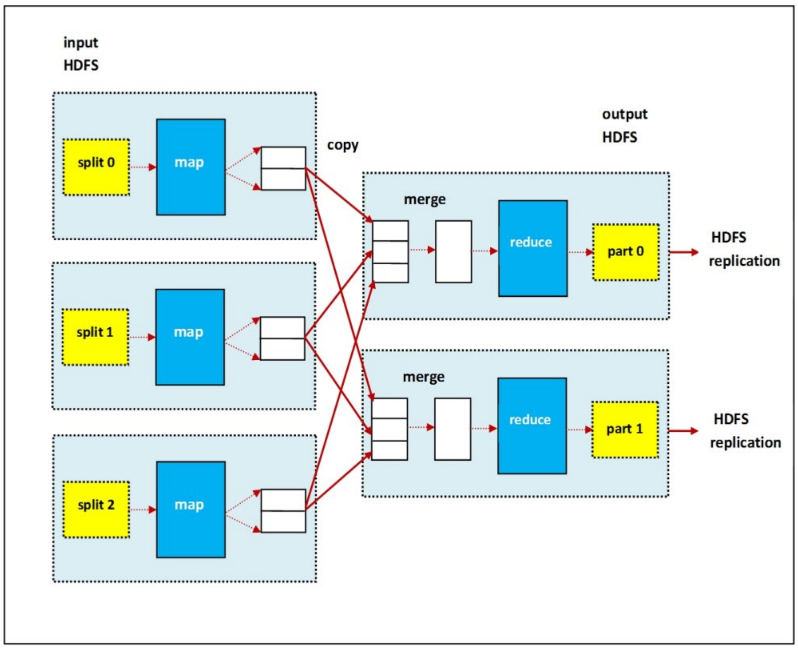

Fig. 1: MapReuce data flow

Performance evaluation is very challenging where virtualization is used to provide a set of computing resources to the customers [4] and as the degree of virtualization (number of VMs running on a single physical box) increases, the complexity gets multiplied. Without any performance degradation a single physical box can host as many as 200 VMs or 35 tiles as per benchmark done by VMware [12]. By default, cloud centers generally host many physical boxes where each can host a number of VMs. Load balancing server is the intermediate arrangement which routes all incoming requests to one of the underlying physical boxes. One or more VMs can be requested by a user at a time. Here, the system allows Batch task arrivals which is similar to the concept of on-demand services provided by Amazon EC2. Different varieties of VM (such as small, medium or large EC2 instances) are available in the Cloud. VMs pricing differs based on the capacity differences of the VMs. So, based on different VM instance types a user can have a variety of Hadoop clusters for the same price range. Moreover, depending upon workload the heterogeneous cluster solution may look lucrative than the homogeneous solution in the range of 26-42% [15].

A number of theoretical studies for finite/infinite, batch arrival queues have been performed with results of varying degrees of complexity in [2, 8, 13]. For a $M[x]=G=m$ queue, an approximation was developed in [10]. An approximate formula is derived by the authors for the steady-state probability of the number of tasks in the system, mean length of the queue and delay probability. But, for batch size larger than three, it gives appreciable amount of error. Performance analysis of cloud centers under batch task arrivals have been discussed in [9]. So, current methods are not suited to be applied for the study of cloud center Overall, existing methods are not well suited for the analysis of cloud centers hosting BigData applications where the number of servers(VMs) are potentially huge, that is, high degree of virtualization and service time distribution is unknown with arrival of requests (Map/Reduce Jobs) in a batch manner.

3 Analytical Model

We model the cloud center as an Markovian $M^{X}/M/c$ queuing system which indicates that tasks (Both MapTasks and ReduceTasks) arrive in batches or groups of random size X with $P(X = k) = gk$ $(k \geq 1)$ and mean batch size $E(X) = \bar{g}$. The inter-arrival times of tasks are exponentially distributed with mean $1/\lambda$. There are c servers (VMs) and service times are independent and exponentially distributed with mean service time $1/\mu$. The service discipline is first-come, first-served (FCFS) and buffer space is infinite. The traffic intensity of the system is $p = \frac{\lambda\bar{g}}{c\mu}$. Each DataNode is allocated a VM and the batch-size is fixed. Below are the characteristics of this model:

- Cloud Centers where tasks arrive in a batch manner, service time of tasks are exponentially distributed and batch-size follows general distribution.
- Provides number of tasks as well as task response time distribution in the system. The probability that immediate service will be provided to a request, response time and waiting time for a request is also provided.
- Cloud Center performance is to be linked with the service time of tasks and batch-size. Impact of larger batch sizes on service time resulting in response time as well as utilization for cloud providers is be measured.

- Impact of Partitioning the incoming requests based on the batch size or service time and then allocating different sub-centers (separate VMs) for processing.

A subset of the model can be represented by three queues *Q*1*;Q*2*;Q*3 and three service stations *S*1*; S*2 and *S*3 where all are connected in parallel and series mode(Figure 2). It is assumed that requests being served through first or second stations join the third queue which is in series-connection to the service stations-*S*1 and *S*2. Requests getting served from *S*1 or *S*2, join the queue-*Q*3. Here *S*1 and *S*2 represents DataNodes representing Mappers whereas *S*3 is a Reducer. It is assumed that the requests (Map-Jobs) in the first two queues are of fixed batch size and the mean arrival rates are not time dependent.

Let us define the steady state probabilities of the system by *Pk*.

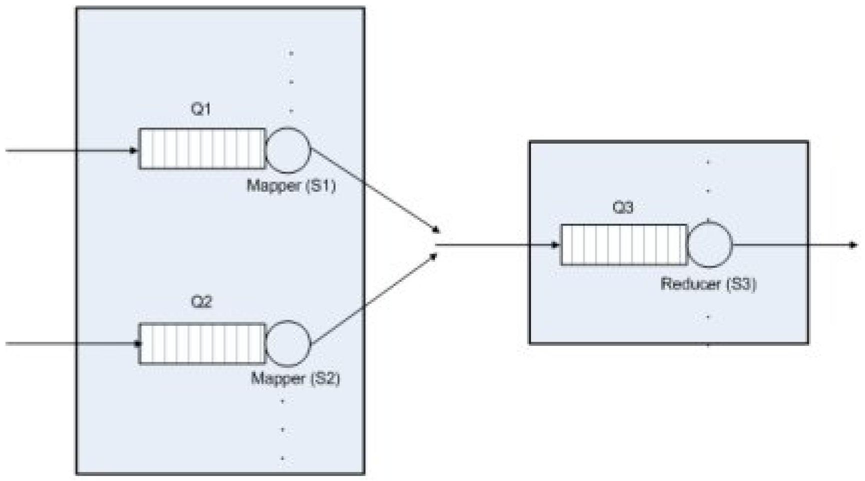

Fig. 2: Schematic diagram representing the queuing model

For the system under consideration, the Chapman-Kolmogorov forward differential equations at the steady state can be represented as

$$\lambda P_0 = \mu P_1 \tag{1}$$

$$(\lambda + i\mu)P_i = (i+1)\mu P_{i+1} + \lambda \sum_{k=0}^{i-1} g_{i-k}P_k, \quad 1 \le i \le c-1, \tag{2}$$

$$(\lambda + c\mu)P_i = c\mu P_{i+1} + \lambda \sum_{k=0}^{i-1} g_{i-k}P_k, \quad i \ge c. \tag{3}$$

Using recursively (1) - (3), we obtain

$$P_1 = \frac{\lambda}{\mu} P_0 \tag{4}$$

$$P_{i+1} = \frac{\lambda + i\mu}{(i+1)\mu} P_i - \frac{\lambda}{(i+1)\mu} \sum_{k=0}^{i-1} g_{i-k} P_k, \ 1 \le i \le c-1, \tag{5}$$

$$P_{i+1} = \frac{\lambda + c\mu}{c\mu} P_i - \frac{\lambda}{c\mu} \sum_{k=0}^{i-1} g_{i-k} P_k, \ i \ge c. \tag{6}$$

Only unknown *P*0 can be obtained using normalization condition as $P_0 = \left(1 + \sum_{i=1}^{\infty} P_i\right)^{-1}$

3.1 Specific batch size distributions

If the arrival batch size follows a geometric distribution, then *P*(*X* = *k*) = $\eta(1-\eta)^{k-1}, \ (k \ge 1), \ 0 < \eta < 1,$ and mean batch size $E(X) = \bar{g} = 1/\lambda.$ Thus, we get

$$P_k = \begin{cases} P_0 \frac{\prod_{r=0}^{k-1} (\theta + r(1-\eta)}{k!}, & 1 \le k \le c, \\ P_c(\rho + (1-\eta)(1-\rho))^{k-c}, & k \ge c+1, \end{cases}$$

where $\theta = \lambda/\mu$ and $\rho = \theta/(c\eta)$.

If the arrival batch size follows a deterministic distribution, then $P(X = k) = 1 \ (k \ge 1)$; and mean batch size $E(X) = \bar{g} = k.$ We obtain P_j as

$$P_j = \frac{\theta}{min(j,c)} \sum_{i=max(0,j-k)}^{j-1} P_i.$$

4 Performance measures

Performance measures are important characteristics of queueing systems as they reflect the efficiency of the model under examination. As the steady-state probabilities are known, the performance measures of the system may be obtained as follows:

- Average number of requests in the queue is

$$L_q = \sum_{i=c+1}^{\infty} (i-c)P_i.$$

- The expected number of busy servers is

$$E(B) = \sum_{k=0}^{c-1} kP_k + \sum_{k=c}^{\infty} cP_k = c - \sum_{k=0}^{c-1} (c-k)P_k$$

- The average number of requests in the system is

$$L_s = \sum_{k=1}^{\infty} kP_k = \sum_{k=1}^{c-1} kP_k + \sum_{k=c}^{\infty} (k-c)P_k + \sum_{k=c}^{\infty} cP_k = L_q + E[B]$$

- Using the Little's formula [5], the average waiting time of a request in the system (*Ws*) and the average waiting time of a request in the queue (*Wq*), respectively, given by

$$W_s = \frac{L_s}{\lambda \bar{g}}, \qquad W_q = \frac{L_q}{\lambda \bar{g}}$$

- The probability that an arriving request has to wait is

$$P(waiting) = \sum_{k=c}^{\infty} P_k$$

5 Numerical results

In this section, numerical results are represented in the form of graphs. It helps managers on taking correct decisions through the qualitative aspects involved in the queueing system considered here through the numerical examples illustrated below. Figure 3 depicts the effect of mean batch size (MBS) on the Traffic-Intensity *p*.

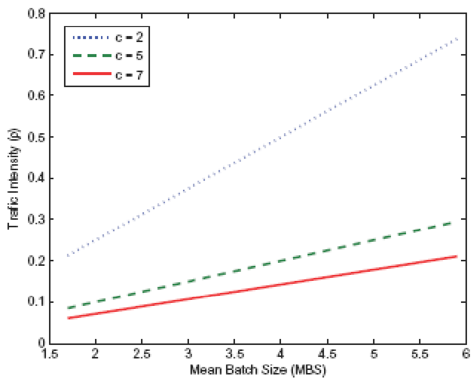

Fig. 3: Impact of Mean-Batch-Size(MBS) on Traffic Intensity *p*

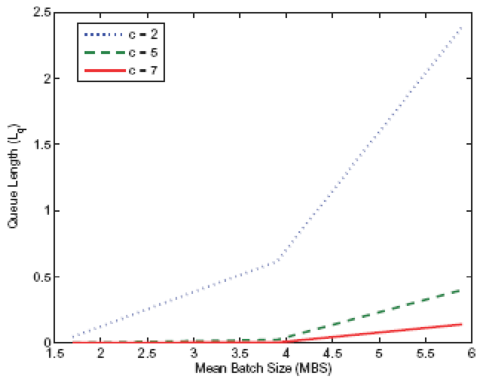

Figure 4: Impact of Mean-Batch-Size ¯*g* on *Lq*

It is evident that as mean batch size increases, _ increases monotonically. Again, it is obvious that the increase in the Traffic-Intensity _ is more when the application has less number of servers. Figure 4 plots the impact of mean batch size on *Lq* for different number of servers, that is, VMs. It can be observed that as the mean batch size increases, the average queue length (*Lq*) increases. But for larger number of servers the average queue length is less and hence making the waiting time less. So, smaller batch size with more number of VMs are preferred. The impact of mean batch size on the mean waiting time in the queue (*Wq*) for various numbers of servers is shown in the Figure 5. It is evident from the figure that *Wq* increases as mean batch size increase. Figure 6 indicates that for more number of servers with higher service rate (_) the *Wq* will be less. Further, with fixed number of servers, the mean waiting time in the queue decreases when the service rate increases. We may setup an admissible service rate and the number of virtual machines to employ servers efficiently.

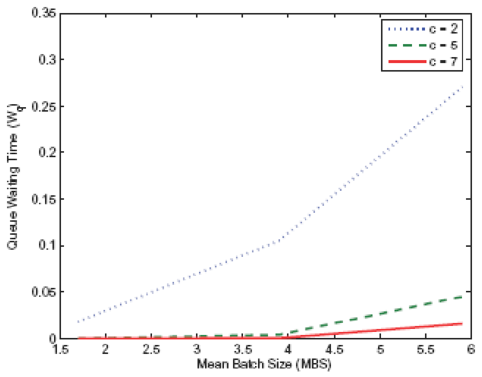

Fig. 5: Impact of mean batch size on *Wq*

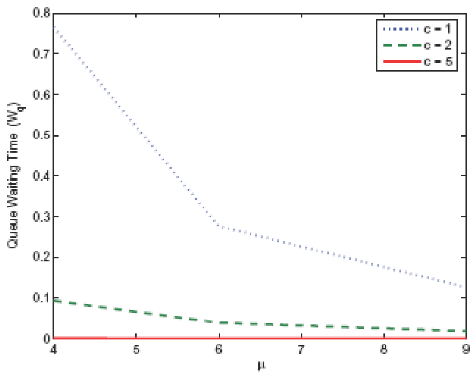

Fig. 6: Impact of _ on *Wq*

Figure 7 depicts the effect of mean batch size on the expected number of busy servers ($E(B)$) with different number of servers. It is seen that as expected number of busy servers increase the mean batch size increase. The expected number of busy servers is more when servers are more. Figure 8 plots the impact of p (traffic intensity) on the mean waiting time in the queue (Wq) under different mean batch sizes. It can be observed that as the traffic intensity increases, the mean waiting time in the queue also increases. We can carefully setup p in the system to ensure the minimum Wq.

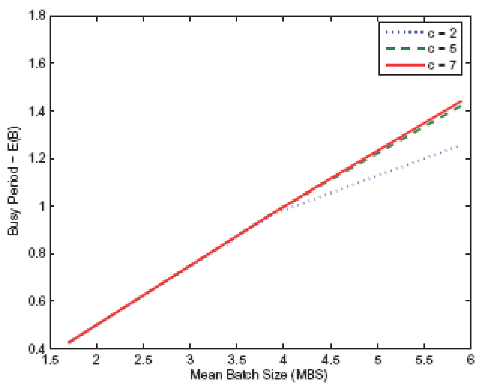

Fig. 7: Impact of mean batch size on $E[B]$

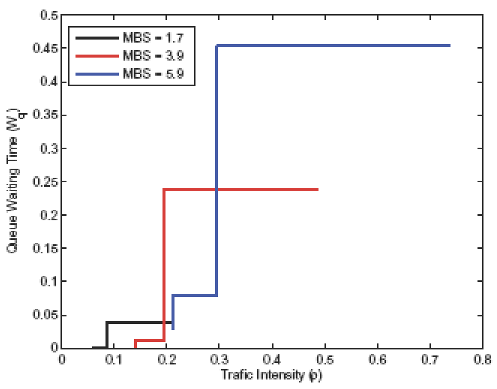

Fig. 8: Impact of _ on Wq

Figure 7 depicts the effect of mean batch size on the expected number of busy servers ($E(B)$) with different number of servers. It is seen that as expected number of busy servers increase the mean batch size increase. The expected number of busy servers is more when servers are more. Figure 8 plots the impact of p (traffic intensity) on the mean waiting time in the queue (Wq) under different mean batch sizes. It can be observed that as the traffic intensity increases, the mean waiting time in the queue also increases. We can carefully setup p in the system to ensure the minimum Wq. From the numerical results, one may determine the effect of system parameters on the performance measures. Figure 9 compares the impact of number of virtual machines on the average system length for various inter batch

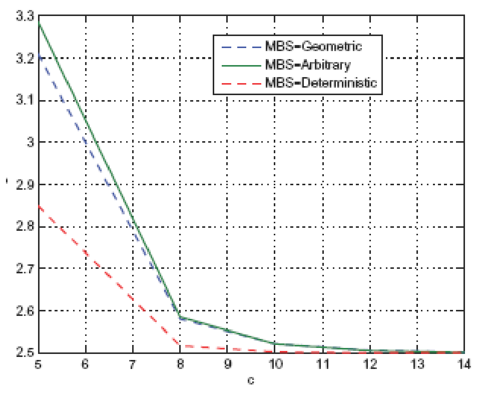

Fig. 9: Impact of c VMs on Ls

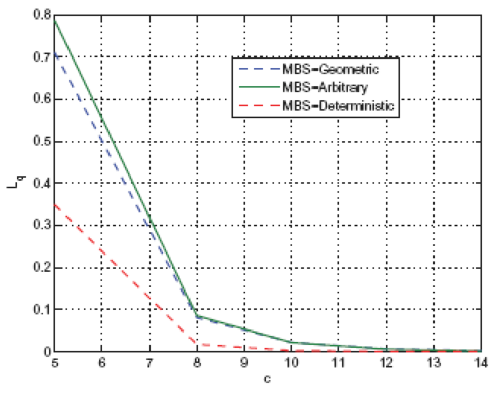

Fig.10: Effect of c VMs on Lq

time distributions with same mean batch size. It is observed that the average system length in the case of deterministic distribution is lower as compared to other distributions. We further observe that the average system length decreases as the number of virtual machines increases and finally reaches to its minimum value for all distributions considered here. The effect of the number of virtual machines on the average queue length is shown in Figure 10. Again one may see that the average queue length decrease as the number of virtual machines increase. As one would intuitively expect, the average queue length in the case of arbitrary distribution is higher as compared to geometric and deterministic distributions.

6 Conclusion

Through Performance evaluation, cloud service providers can tailor their SLAs so that mutually agreed benchmark can be achieved for their customers. This paper depicts a analytical model for cloud centers providing MapReduce as service, that is, batch arrival of Mapper/Reducer requests with no rejection policy. In order to map correctly with the cloud centers, the service time for each task contained within batch-tasks and batch size is generally distributed with higher degree of virtualization, that is, large number of servers (VMs) involved. Important performance parameters like mean queue length, mean number of tasks in the system, mean waiting and mean response time are accurately calculated by using the proposed method here. As per the findings we can conclude about the cloud centers allowing broadly varying service times may have longer waiting time in the queue with less chance of acquiring immediate service and hence less utilized servers (VMs), in comparison with the equivalent cloud centers which deal with fixed types of tasks. Moreover, when we have bigger batch size that will lead to more waiting time and hence less utilization of resources and thereby making more operational cost for the cloud service provider.

.

References

[1] D. Borthakur. The hadoop distributed file system: Architecture and design. *Hadoop Project Website*, 11(2007):21, 2007.

[2] M. Cromie, M. Chaudhry, W. Grassmann. Further results for the queueing system *MX=M=c*. *Journal of the Operational Research Society*, strony 755–763, 1979.

[3] J. Dean, S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[4] J. Fu, W. Hao, M. Tu, B. Ma, J. Baldwin, F. B. Bastani. Virtual services in cloud computing. *Services (SERVICES-1), 2010 6th World Congress on*, strony 467–472. IEEE, 2010.

[5] D. Gross, C. M. Harris. *Fundamentals of queueing theory*. 1998.

[6] D. Guide. Amazon elastic mapreduce. 2010.

[7] B. Igou. User survey analysis: Cloud-computing budgets are growing and shifting; traditional it services providers must prepare or perish. *Gartner Report*, 2010.

[8] I. W. Kabak. Blocking and delays in $M(n)=M=c$ bulk queuing systems. *Operations Research*,16(4):830–840, 1968.

[9] H. Khazaei, J. Misic, V. Misic. Performance of cloud centers with high degree of virtualization under batch task arrivals. *Parallel and Distributed Systems, IEEE Transactions on*, 24(12):2429– 2438, 2013.

[10]T. Kimura, T. Ohsone. A diffusion approximation for an $M=G=m$ queue with group arrivals. *Management Science*, 30(3):381–388, 1984.

[11]R. L¨ammel. Googles mapreduce programming model revisited. *Science of computer programming*, 70(1):1–30, 2008.

[12]V. Makhija, B. Herndon, P. Smith, L. Roderick, E. Zamost, J. Anderson. Vmmark: A scalable benchmark for virtualized systems. *VMware Inc, CA, Tech. Rep. VMware-TR-2006-002*, 2006.

[13]D. R. Manfield, P. Tran-Gia. Analysis of a finite storage system with batch input arising out of message packetization. *Communications, IEEE Transactions on*, 30(3):456–463, 1982.

[14]T. White. *Hadoop: The de_nitive guide*. O'Reilly Media, Inc., 2012.

[15]Z. Zhang, L. Cherkasova, B. T. Loo. Exploiting cloud heterogeneity to optimize performance and cost of mapreduce processing. *ACM SIGMETRICS Performance Evaluation Review*, 42(4):38–50, 2015.