symmetry

MDPI

Article

An Efficient and Energy-Aware Cloud Consolidation Algorithm for Multimedia Big Data Applications

JongBeom Lim [1], HeonChang Yu [2] and Joon-Min Gil [3,*]

[1] Department of Game & Multimedia Engineering, Korea Polytechnic University, Siheung-si, Gyeonggi-do 15073, Korea; jblim@kpu.ac.kr
[2] Department of Computer Science & Engineering, Korea University, Seoul 02841, Korea; yuhc@korea.ac.kr
[3] School of Information Technology Engineering, Daegu Catholic University, Gyeongsan-si, Gyeongsangbuk-do 38430, Korea
* Correspondence: jmgil@cu.ac.kr

Received: 14 August 2017; Accepted: 1 September 2017; Published: 6 September 2017

Abstract: It is well known that cloud computing has many potential advantages over traditional distributed systems. Many enterprises can build their own private cloud with open source infrastructure as a service (IaaS) frameworks. Since enterprise applications and data are migrating to private cloud, the performance of cloud computing environments is of utmost importance for both cloud providers and users. To improve the performance, previous studies on cloud consolidation have been focused on live migration of virtual machines based on resource utilization. However, the approaches are not suitable for multimedia big data applications. In this paper, we reveal the performance bottleneck of multimedia big data applications in cloud computing environments and propose a cloud consolidation algorithm that considers application types. We show that our consolidation algorithm outperforms previous approaches.

Keywords: cloud consolidation; virtual machine; big data; multimedia application; cloud computing

1. Introduction

The proliferation of multimedia big data and sharing of large amounts of information for various multimedia applications have brought an urgent need to develop efficient methods of processing them [1]. The most visible sources of such multimedia big data are social media/networks [2] and mobile services [3]. In the big data era, multimedia big data has become an active and interdisciplinary research field since it has introduced challenges and opportunities in the existing computing environments including cloud computing.

To deal with the large size of the data, distributed high performance computing (HPC) systems are required to process multimedia big data. Among HPC systems, cloud computing has received great attention during the last decade as an emerging paradigm [4]. However, the characteristics of multimedia big data make it very difficult to process the data efficiently in cloud computing environments. In addition, the virtual machine (VM) in cloud computing environments introduces some degree of performance degradation, causing frequent VM exits [5]. Furthermore, the type of multimedia big data application is data-intensive: this means that the storage systems of cloud computing environments also affect the performance [6].

As free and open-source cloud computing software platforms for infrastructure as a service (IaaS) have become available, community and private clouds have widely been built around the world. Openstack is one of the most popular open-source software platform for IaaS. However, compared to the public clouds such as Amazon elastic compute cloud (EC2) and Google compute engine, community and private clouds are more commonly hosted on commodity hardware. In this regard, to process multimedia big data in commodity and private clouds, an efficient cloud consolidation (whose aim

is to make more efficient use of computer resources and prevent servers from being under-utilized) is required.

While cloud computing has been considered as an efficient solution for processing parallel applications and CPU intensive workloads [7], cloud resource consolidation frameworks for multimedia big data have not been fully developed. In this context, combining cloud infrastructure systems with multimedia big data is a relatively new field of computer science since cloud computing was developed by major IT providers whose major objective was to develop high performance computing solutions using the techniques of grid computing and utility computing [4,8].

More specifically, past HPC systems mainly targeted traditional scientific workloads and applications. These scientific applications had high arithmetic intensity, locality unlike multimedia big data, and were easy to parallel into distributed systems [9]. In other words, in order to process multimedia big data applications efficiently, it is necessary to take the characteristics of multimedia big data into account.

In this paper, we reveal the performance bottleneck of multimedia big data applications in cloud computing environments and propose a cloud consolidation algorithm that assigns tasks and VMs considering the tradeoff between the number of VMs and input/output (I/O) bottlenecks. The proposed consolidation algorithm consists of three parts: pre-processing, VM monitoring, and task and VM assignment. All of these parts are correlated to perform the cloud consolidation for multimedia big data.

The remainder of the paper is organized as follows: In Section 2, we describe the research motivation and related work. The proposed cloud consolidation algorithm for multimedia big data application is presented in Section 3. Section 4 presents performance evaluation with realistic scenarios. Finally, Section 5 concludes the paper.

2. Motivation and Related Work

In this section, we describe our research motivation for multimedia big data applications and detail related work on cloud consolidation. This section also highlights the problem for processing multimedia big data applications in cloud computing environments.

2.1. Research Motivation

Compared to text-based big data applications, multimedia big data applications have different characteristics [10] with respect to organizing unstructured and heterogeneous data, dealing with cognition and understanding complexity, addressing real-time and quality-of-service requirements, and ensuring scalability and computing efficiency.

We found that I/O bottlenecks exist when multimedia big data applications are performed on cloud computing environments as depicted in Figure 1. If a cloud computing infrastructure is configured with the Openstack block storage service (Cinder), which provides persistent block-level storage devices for VMs, the Cinder node cannot guarantee the service level agreement (SLA) as the number of compute nodes increase because multimedia big data applications are data intensive workloads. If the administrator configures the cloud computing infrastructure with block storage enabled on compute nodes, I/O bottlenecks still exist when compute nodes embrace a large number of VMs.

Figure 2 shows the execution time of multimedia big data applications that are performed in a cloud computing environment. Each VM performs a video encoding application (H.264) for a 5 GB file (personally recorded 20 min 1080 p video file). When the number of VMs is 1, the execution time is about 165 s. However, when the number of VMs is 2 and 3, the execution time of multimedia big data applications is about 462 and 737 s, respectively. The execution time is more than 2 and 3 times slower when the number of VMs is 2 and 3, respectively, compared to that when the number of VMs is 1. Note that when the number of VMs is greater than 1, the execution time is averaged by the number of VMs.

These problems motivate us to develop a cloud consolidation algorithm for multimedia big data applications. Our cloud consolidation algorithm considers the tradeoff between the number of VMs and I/O bottlenecks on the cloud infrastructure and assigns tasks (to VMs) and VMs (to physical machines) in an efficient manner, while reducing total execution time.

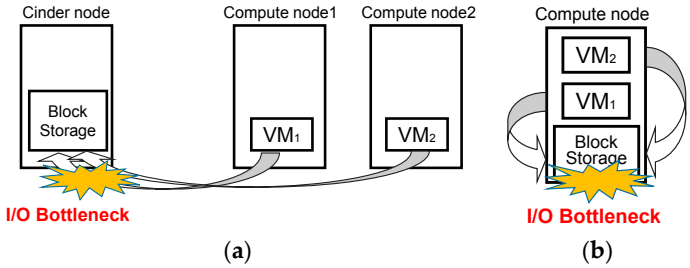


Figure 1. I/O bottleneck when multimedia big data applications are performed on cloud computing environments. (**a**) I/O bottleneck with Cinder node; (**b**) I/O bottleneck with block storage.

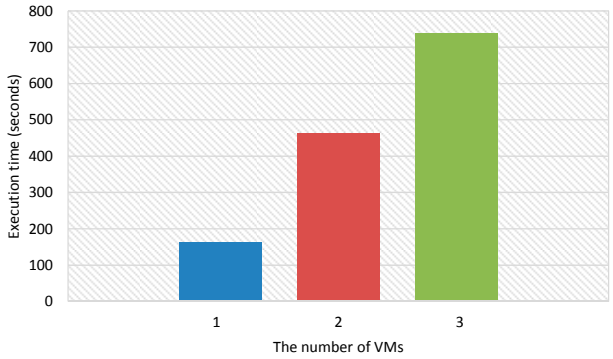


Figure 2. Execution time when multimedia big data applications are performed on cloud computing environments.

2.2. Related Work

With Internet and its applications becoming ubiquitous, huge amounts of multimedia data are being produced. This is forming a unique kind of big data: multimedia big data [10]. Multimedia big data introduces new kinds of multimedia services. Some of multimedia big data applications include digital image processing [11], audio/video recommendations [12], video annotation [13], online advertisements [14], video classification [15], and multimedia delivery for mobile devices [16]. Therefore, cloud computing gains wide acceptance for processing multimedia big data since it provides a number of benefits, including reduced IT costs, flexibility, and increased collaboration [17].

Since multimedia big data applications have distinctive characteristics compared to traditional big data applications, many researchers have studied some of the problems in big data computing in cloud computing environments. Multimedia big data applications and services are typically real-time, therefore, an efficient cloud consolidation algorithm for processing multimedia big data is required to address quality of experience (QoE) [18].

Optimizing the performance of big data applications has been extensively studied in cloud computing environments. Simmhan et al. [19] describe the dynamic demand response (D^2R): a cloud-based software platform for big data analytics in smart grids. The platform offers an adaptive information integration pipeline for ingesting dynamic data and the pipeline stages are as follows: data transport from remote data sources, data parsing to interpret the structure, semantic annotation to enrich the data context, and data storage for information persistence.

Keke Gai and his colleagues [20] proposed the cost-aware heterogeneous cloud memory model (CAHCM) for multimedia data allocation, to provide a high performance cloud-based heterogeneous

memory service. Its dynamic data allocation advance (2DA) algorithm considers communication costs, data move operating costs, energy performance, and time constraints. The idea of the algorithm is to enable CAHCM to mitigate the big data processing to remote facilities by using cloud-based memories.

Jayasena et al. [21], developed the multimedia big data analyzing architecture and resource allocation scheme with three layers (service layer, platform layer and infrastructure layer) on top of a MapReduce framework running on a Hadoop distributed file system (HDFS). The aim of the architectural design is to reduce the time for transcoding large amounts of data into specific formats, and to achieve a minimal response time for allocating VMs with the ant colony optimization (ACO) algorithm in the infrastructure layer.

Taotao Wu and his colleagues [18] introduced a problem called the optimal deployment problem (ODP) and the influence of local memory to migrate large-scale media streaming applications to the cloud. By exploiting the local memory, it can decrease total reservation bandwidth from cloud data centers and enhance the utilization of bandwidth.

For object detection in multimedia big data, Guo et al. [22] considered the unbalanced computation capacity between a mobile center cloud and mobile edge sites. To shift large storage burden into the cloud, they performed dictionary learning to obtain the sparse representation in the pixel domain. By taking advantage of low latency of the mobile edge computing ends, the computation cost and storage cost of the proposed detection process can be reduced. VM or task migration (network transfer time) [23,24], mobile cloud computing [25], and the computing model for knowledge discovery [26] are also interesting topics in the field. However, we focus on the verification of the proposed algorithm and will integrate a genetic algorithm [27] to enhance the performance. Hence, we leave further exploration of this possibility for future work.

While above studies have shown that there is a strong relationship between multimedia big data and cloud computing, none of them are directly related to cloud consolidation and able to solve the fundamental problem for multimedia big data application in cloud computing environments, that is, the cloud task and VM assignment. Hence, our approach differs from these previous studies in that we reveal the performance bottleneck of multimedia big data applications in cloud computing environments and propose a cloud consolidation algorithm that assigns tasks and VMs considering the tradeoff between the number of VMs and I/O bottlenecks.

3. The Proposed Consolidation Algorithm

In this section, we detail our proposed consolidation algorithm for big data applications in cloud computing environments. The proposed consolidation algorithm utilizes the cloud system's information about hosts and VMs. Once a user submits cloud tasks to the cloud system, our consolidation algorithm dynamically performs the consolidation algorithm for mapping between cloud tasks and resources. Then, it assigns the cloud tasks to VMs according to the mapping information.

3.1. System Model

We consider a private cloud computing infrastructure that consists of a number of physical machines with commodity hardware. Therefore, many VMs can be provisioned with the help of virtualization technology. Cloud users can submit their multimedia big data applications (tasks) to the cloud computing infrastructure and a consolidation algorithm is in charge of assigning tasks and VMs.

Figure 3 shows the system model of the proposed consolidation algorithm. There are two distinct modules: preprocessing module and resource management module. The preprocessing module is in charge of retrieving task information and calculating the processing cost for cloud tasks. The results of the preprocessing module are used in the resource management module.

The resource management module performs procedures related to VM monitoring, VM provisioning, and task assignment. The VM monitoring procedure is triggered at regular intervals so that our consolidation algorithm uses the up-to-date resource information of the cloud system. Based on VM monitoring information, our consolidation algorithm generates mapping information

between cloud tasks and resources. With results of the resource management module, cloud tasks can be deployed and the deployment information is stored in the database for future reference. When the same or similar tasks are submitted, the preprocessing step can be skipped with the stored history information. Finally, the cloud returns the results to the user.

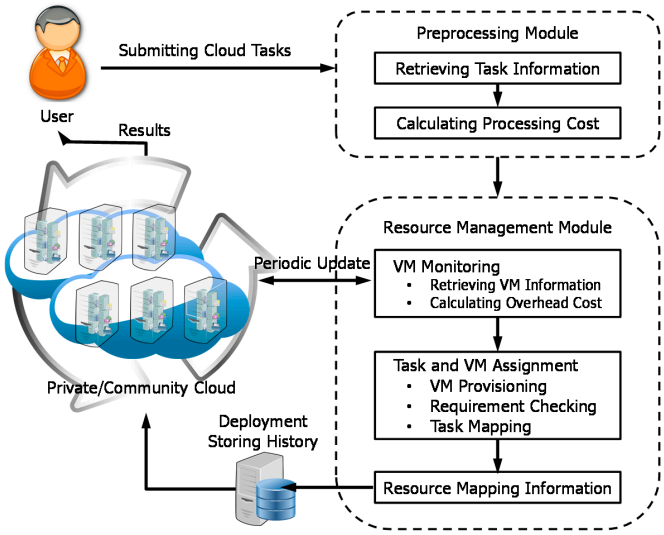


Figure 3. System model.

3.2. Preprocessing

Algorithm 1 shows the preprocessing algorithm before task and VM assignment. The input is a set of cloud tasks submitted by cloud users and the output is an ordered set of cloud tasks. The preprocessing step is necessary to consider I/O bottlenecks. In other words, after the preprocessing step is performed, the most data intensive cloud task can be scheduled first to minimize I/O bottlenecks and execution time. More specifically, as the tasks are sorted by processing cost, the most data intensive task can be allocated first to the most powerful VM. Note that in previous research, the tasks are allocated with first in first out (FIFO) or random policies.

For each task in the task set, the preprocessing algorithm retrieves task information and calculates the processing cost considering datasets and application types. Note that task information includes job ID, user, task type, submit time, start time, end time, exit status, suspend time, wall time, CPU time, I/O time, etc. Based on task information, the cost value is measured and the range of the cost is (0, 1). For example, if the average CPU utilization is 50%, the cost for CPU is 0.5. Then, it stores the information and sorts the tasks in descending order of the calculated cost information. The sorted task set is used for the task and VM assignment algorithm.

Algorithm 1. The Preprocessing Algorithm

Input: T_i, where $\forall i \in \{1, 2, \ldots, n\}$
Output: Ordered set of T_i
1: **begin**
2: **for each** $T_i \in TaskSet$
3: Retrieve task information;
4: Calculate processing cost;
5: $TempTaskSet \leftarrow (T_i, calculated\_cost)$;
6: **end for**
7: Sort *TempTaskSet* in descending order by *calculated_cost*;
8: Return TempTaskSet;
9: **end**

3.3. Task and VM Assignment Algorithm

To effectively manage our cloud consolidation procedure, it is essential to gain resource-monitoring information. Algorithm 2 shows the VM monitoring algorithm and this algorithm can be periodically called or performed on demand. The procedure is similar to that of the preprocessing algorithm. The VM monitoring algorithm maintains the overhead cost based on VM information. Then, it returns an ordered set of resources in ascending order by the overhead cost.

Once the preprocessing process is successfully performed, the next step is to assign tasks and VMs. Algorithm 3 shows the task and VM assignment algorithm. The input is the ordered task set and ordered resource set. The output is mapping information between cloud tasks and resource information. For each task from the ordered task set, it extracts the first element from the ordered resource set. Then, it checks whether the resource can run the tasks. If the resource does not meet the requirement, it triggers the VM provisioning task and assigns the task to the newly provisioned VM.

Details of the VM provisioning task are as follows: First, it searches physical machines suitable for VM provisioning. Second, it commands the selected physical machine to generate a new VM. Lastly, it informs the cloud resource manager for the newly provisioned VM. Then, the algorithm assigns T_i to R_j. For the next iteration, it undertakes the monitoring task and removes the assigned task from the ordered task set.

Algorithm 2. The VM Monitoring Algorithm

Input: R_j, where $\forall j \in \{1, 2, \ldots, m\}$
Output: Ordered set of R_j
begin
  for each $R_j \in ResourceSet$
    Retrieve resource information;
    Calculate overhead cost
    $TempResourceSet \leftarrow (R_j, overhead\_cost)$;
  end for
  Sort *TempResourceSet* in ascending order by *overhead_cost*;
  Return TempResourceSet;
end

Algorithm 3. The Task and VM Assignment Algorithm

Input: *OrderedTaskSet*, *OrderedResourceSet*
Output: map (T_i, R_j), where $\forall i \in \{1, 2, \ldots, n\}$, $\forall j \in \{1, 2, \ldots, m\}$
begin
  for each $T_i \in OrderedTaskSet$
    $R_j \leftarrow$ Extract the first element from *OrderedResourceSet*
    if R_j does not meet the requirement **then**
      Trigger VM provisioning task;
      $R_j \leftarrow$ Get newly provisioned resource information;
    end if
    Map T_i to R_j;
    Trigger monitoring task for the next iteration
    $OrderedTaskSet = OrderedTaskSet - T_i$;
  end for
  return map (T_i, R_j);
end

4. Performance Evaluation of the Algorithm

In this section, we present experimental results that demonstrate the performance of our cloud consolidation algorithm for multimedia big data applications by managing the tradeoff between the

number of VMs and I/O bottlenecks. As input, we use real task traces (Intel Netbatch logs [28]) and artifact task logs for big data applications. Note that the artifact task logs are configured for video encoding applications. The Intel Netbatch logs are chosen as they were generated by multiple physical clusters with different numbers of nodes (tens of thousands) and contain information about completed jobs with various properties (e.g., CPU, the number of cores, memory, processing time, and job's type). Therefore, we can configure multimedia big data applications for cloud tasks by adding I/O and data processing information.

For experiments, we use a discrete event simulator for iterative and extensive evaluations. We consider 50 hosts (physical machines) and 100 VMs running in the cloud infrastructure and assume that a cloud user submits 100 cloud tasks and the task type is multimedia big data applications unless specified otherwise. For comparison, we implement the elementary consolidation technique that assigns VMs and tasks with a basic rule (i.e., it assigns tasks to VMs in FIFO order).

Figure 4 shows performance results for execution time, mean time before shutdown, and standard deviation time before shutdown (without live migration). Note that the execution time is the elapsed time between when a user submits cloud tasks and the cloud finishes all the cloud tasks.

The execution time for the basic approach is about 1,748,100 s and that for our proposed approach is about 1,060,200 s. The reduction of execution time is about 40% compared to the basic approach, which confirms that our consolidation algorithm is efficient and can manage the tradeoff between the number of VMs and I/O bottlenecks for multimedia big data applications.

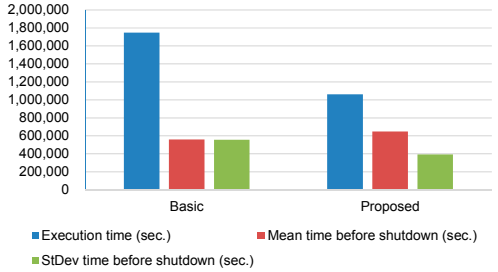


Figure 4. Performance results for execution time, mean time before shutdown, and standard deviation time before shutdown (without live migration).

Figure 5 depicts performance results for execution time, mean time before shutdown, and standard deviation time before shutdown (with live migration). Note that the Y-axis is represented on a logarithmic scale. Comparing to Figure 4 (without live migration), mean time before shutdown and standard deviation time before shutdown are greatly reduced (for both Basic and Proposed). Nonetheless, the proposed consolidation algorithm results in more than 4 times less mean time before shutdown than the basic approach. With respect to standard deviation time before shutdown, the proposed consolidation algorithm outperforms the basic approach by more than 10 times.

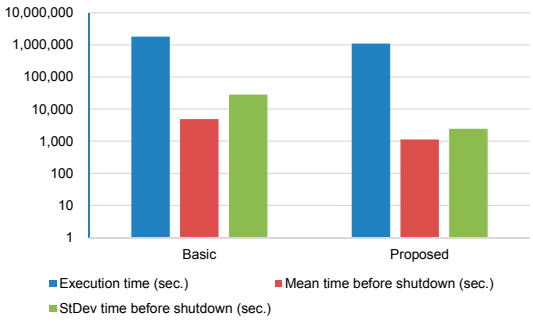
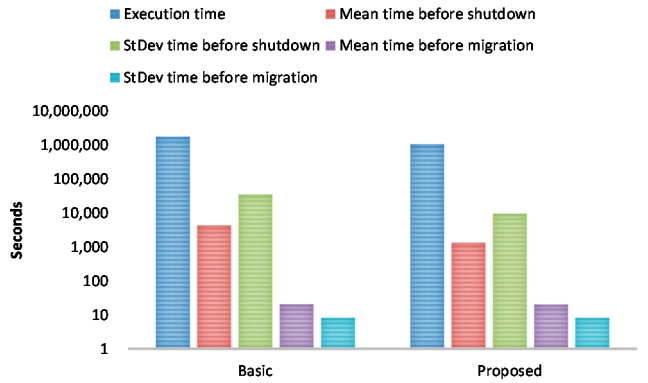


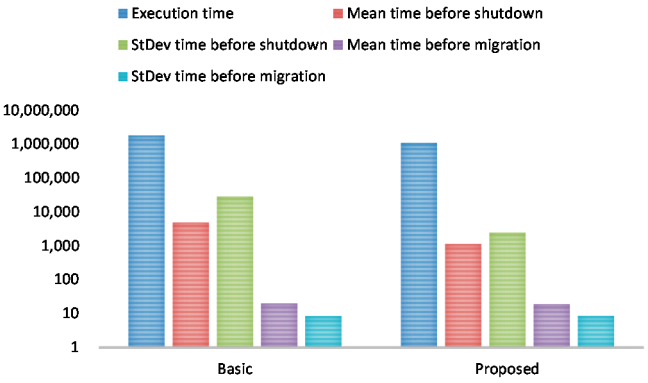
Figure 5. Performance results for execution time, mean time before shutdown, and standard deviation time before shutdown (with live migration).

To show the scalability of the proposed consolidation algorithm in terms of the number of VMs, we varied the number of VMs from 50 to 200. Figure 6 shows performance results for execution time, mean time before shutdown, standard deviation time before shutdown, mean time before migration, and standard deviation time before migration when the number of VMs is 50, 100, 150, and 200.

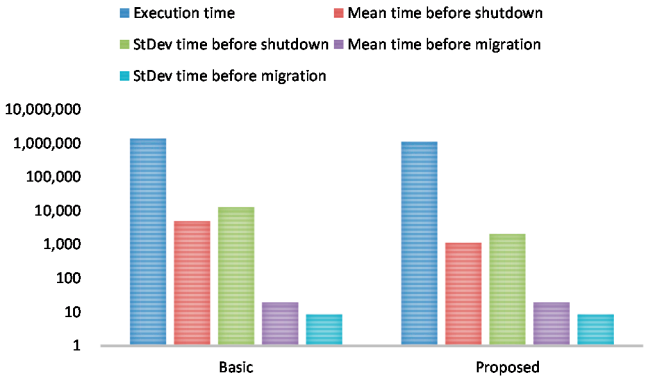
With our consolidation algorithm, the execution time is reduced by 40% on average compared to the basic approach. For mean time before shutdown and standard deviation time before shutdown, the proposed consolidation algorithm always outperforms the basic approach. With respect to mean time before migration and standard deviation time before migration, the results are comparable between the proposed approach and the basic approach.



(**a**)

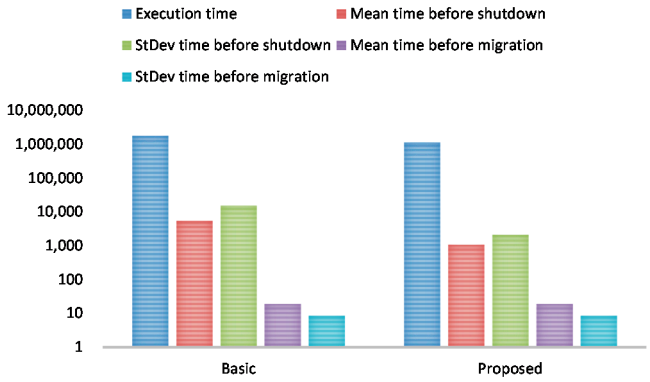


(**b**)



(**c**)

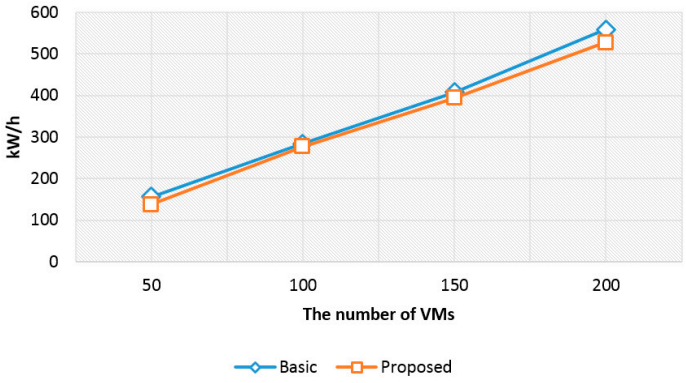
Figure 6. *Cont.*



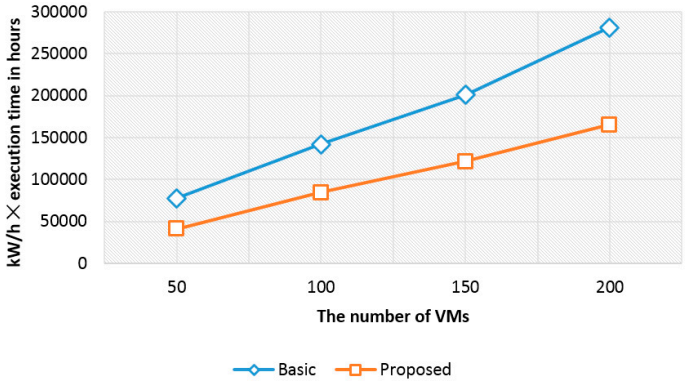
(**d**)

Figure 6. Performance results for execution time, mean time before shutdown, standard deviation time before shutdown, mean time before migration, and standard deviation time before migration. (**a**) The number of virtual machines (VMs): 50; (**b**) The number of VMs: 100; (**c**) The number of VMs: 150; (**d**) The number of VMs: 200.

Reducing energy consumption is also important in cloud computing environments, as it relates to the cost to run cloud tasks [29]. To validate the energy efficiency of the proposed consolidation algorithm, we measured the energy consumption as in Figure 7. The kW/h is comparable, but our proposed approach slightly outperforms the basic approach (cf. Figure 7a). The basic approach consumes more energy (about 13%) than our approach when the number of VMs is 50. When the number of VMs is 100, 150, and 200, the proposed approach consumes less energy than the basic approach about 3%, 4%, and 6%, respectively.



(**a**)



(**b**)

Figure 7. Performance results for kW/h. (**a**) kW/h; (**b**) kW/h × execution time in hours.

Since our proposed consolidation algorithm reduces the execution time, kW/h × execution time in hours can also be reduced (cf. Figure 7b). The result of our proposed approach is about 52% compared to the basic approach when the number of VMs is 50. When the number of VMs is 100, 150, and 200, the result of our proposed approach is about 59%, 60%, and 58%, respectively, compared to the basic approach. Our approach not only reduces the execution time for cloud tasks, but saves energy by balancing the I/O bottleneck in cloud computing environments.

Table 1 shows performance results for SLA violation. The SLA violation stems from live migration, and performance degradation due to live migration that is about 0.34%. Regardless of the number of VMs, our approach generates less than 0.1% of SLA violation, which is about 10% less than that of the basic approach. This also confirms the effectiveness of our consolidation algorithm.

Table 1. Performance results for service level agreement (SLA) violation.

The Number of VMs	Basic	Proposed
50	0.062%	0.056%
100	0.084%	0.075%
150	0.102%	0.087%
200	0.105%	0.097%

5. Conclusions

In this paper, we revealed I/O bottlenecks of multimedia big data applications running on cloud computing infrastructures and proposed a cloud consolidation algorithm that assigns tasks and VMs considering the tradeoff between the number of VMs and I/O bottlenecks on cloud infrastructure. The performance results show that the execution time of cloud tasks is reduced by about 40% when our consolidation algorithm is used with respect to the basic approach. Future work includes energy aware scheduling techniques with live migration for multimedia big data applications.

Acknowledgments: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2016R1D1A3B03933370 and NRF-2015R1D1A1A01061373).

Author Contributions: All the authors contributed equally to the work. All authors read and approved the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chen, M.; Mao, S.; Liu, Y. Big data: A survey. *Mob. Netw. Appl.* **2014**, *19*, 171–209. [CrossRef]
2. Alaimo, C.; Kallinikos, J. Computing the everyday: Social media as data platforms. *Inf. Soc.* **2017**, *33*, 175–191. [CrossRef]
3. Su, Z.; Xu, Q.; Qi, Q. Big data in mobile social networks: A qoe-oriented framework. *IEEE Netw.* **2016**, *30*, 52–57. [CrossRef]
4. Buyya, R.; Yeo, C.S.; Venugopal, S.; Broberg, J.; Brandic, I. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.* **2009**, *25*, 599–616. [CrossRef]
5. Agesen, O.; Mattson, J.; Rugina, R.; Sheldon, J. Software techniques for avoiding hardware virtualization exits. In Proceedings of the 2012 USENIX Conference on Annual Technical Conference, Boston, MA, USA, 13–15 June 2012; USENIX Association: Berkeley, CA, USA, 2012; pp. 373–385.
6. Karakoyunlu, C.; Chandy, J.A. Exploiting user metadata for energy-aware node allocation in a cloud storage system. *J. Comput. Syst. Sci.* **2016**, *82*, 282–309. [CrossRef]
7. Escheikh, M.; Barkaoui, K.; Jouini, H. Versatile workload-aware power management performability analysis of server virtualized systems. *J. Syst. Softw.* **2017**, *125*, 365–379. [CrossRef]
8. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; et al. A view of cloud computing. *Commun. ACM* **2010**, *53*, 50–58. [CrossRef]

9. Dongarra, J.J.; van der Steen, A.J. High-performance computing systems: Status and outlook. *Acta Numer.* **2012**, *21*, 379–474. [CrossRef]
10. Zhu, W.; Cui, P.; Wang, Z.; Hua, G. Multimedia big data computing. *IEEE Multimed.* **2015**, *22*, 96–c3. [CrossRef]
11. Zhang, W.; Zhou, R.; Zou, Y. Self-adaptive and bidirectional dynamic subset selection algorithm for digital image correlation. *J. Inf. Process. Syst.* **2017**, *13*, 305–320.
12. Alnusair, A.; Zhong, C.; Rawashdeh, M.; Hossain, M.S.; Alamri, A. Context-aware multimodal recommendations of multimedia data in cyber situational awareness. *Multimed. Tools Appl.* **2017**, 1–21. [CrossRef]
13. Viana, P.; Pinto, J.P. A collaborative approach for semantic time-based video annotation using gamification. *Hum. Cent. Comput. Inf. Sci.* **2017**, *7*, 13. [CrossRef]
14. Cheng, Z.Q.; Wu, X.; Liu, Y.; Hua, X.S. Video ecommerce++: Toward large scale online video advertising. *IEEE Trans. Multimed.* **2017**, *19*, 1170–1183. [CrossRef]
15. Koutsakis, P.; Spanou, I.; Lazaris, A. Video scene identification and classification for user-tailored qoe in geo satellites. *Hum. Cent. Comput. Inf. Sci.* **2017**, *7*, 15. [CrossRef]
16. Ciubotaru, B.; Muntean, C.H.; Muntean, G.M. Mobile multi-source high quality multimedia delivery scheme. *IEEE Trans. Broadcast.* **2017**, *63*, 391–403. [CrossRef]
17. Yu, Y.; Miyaji, A.; Au, M.H.; Susilo, W. Cloud computing security and privacy: Standards and regulations. *Comput. Stand. Interfaces* **2017**, *54*, 1–2. [CrossRef]
18. Wu, T.; Dou, W.; Wu, F.; Tang, S.; Hu, C.; Chen, J. A deployment optimization scheme over multimedia big data for large-scale media streaming application. *ACM Trans. Multimed. Comput. Commun. Appl.* **2016**, *12*, 1–23. [CrossRef]
19. Simmhan, Y.; Aman, S.; Kumbhare, A.; Liu, R.; Stevens, S.; Zhou, Q.; Prasanna, V. Cloud-based software platform for big data analytics in smart grids. *Comput. Sci. Eng.* **2013**, *15*, 38–47. [CrossRef]
20. Gai, K.; Qiu, M.; Zhao, H. Cost-aware multimedia data allocation for heterogeneous memory using genetic algorithm in cloud computing. *IEEE Trans. Cloud Comput.* **2017**, *PP*, 1. [CrossRef]
21. Jayasena, K.P.N.; Li, L.; Xie, Q. Multi-modal multimedia big data analyzing architecture and resource allocation on cloud platform. *Neurocomputing* **2017**, *253*, 135–143. [CrossRef]
22. Guo, J.; Song, B.; Richard Yu, F.; Yan, Z.; Yang, L.T. Object detection among multimedia big data in the compressive measurement domain under mobile distributed architecture. *Future Gener. Comput. Syst.* **2017**, *76*, 519–527. [CrossRef]
23. De Maio, V.; Prodan, R.; Benedict, S.; Kecskemeti, G. Modelling energy consumption of network transfers and virtual machine migration. *Future Gener. Comput. Syst.* **2016**, *56*, 388–406. [CrossRef]
24. Rathod, S.B.; Reddy, V.K. Ndynamic framework for secure vm migration over cloud computing. *J. Inf. Process. Syst.* **2017**, *13*, 476–490.
25. Gai, K.; Qiu, M.; Zhao, H.; Tao, L.; Zong, Z. Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing. *J. Netw. Comput. Appl.* **2016**, *59*, 46–54. [CrossRef]
26. Finogeev, A.G.; Parygin, D.S.; Finogeev, A.A. The convergence computing model for big sensor data mining and knowledge discovery. *Hum. Cent. Comput. Inf. Sci.* **2017**, *7*, 11. [CrossRef]
27. Tao, F.; Feng, Y.; Zhang, L.; Liao, T.W. CLPS-GA: A case library and pareto solution-based hybrid genetic algorithm for energy-aware cloud service scheduling. *Appl. Soft Comput.* **2014**, *19*, 264–279. [CrossRef]
28. Shai, O.; Shmueli, E.; Feitelson, D.G. Heuristics for resource matching in intel's compute farm. In Proceedings of the Job Scheduling Strategies for Parallel Processing: 17th International Workshop—JSSPP 2013, Boston, MA, USA, 24 May 2013; Desai, N., Cirne, W., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; pp. 116–135.
29. Procaccianti, G.; Lago, P.; Bevini, S. A systematic literature review on energy efficiency in cloud software architectures. *Sustain. Comput.Inf. Syst.* **2015**, *7*, 2–10. [CrossRef]