Journal of Theoretical and Applied Information Technology
29th February 2016. Vol.84. No.3
© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ENHANCEMENT OF CLOUD PERFORMANCE AND STORAGE CONSUMPTION USING ADAPTIVE REPLACEMENT CACHE AND PROBABILISTIC CONTENT PLACEMENT ALGORITHMS

[1]AHMED SALIH MAHDI, [2]RAVIE CHANDREN MUNIYANDIH

[1]Master student, Department of Network Technology, Faculty of Science and Technology, Universiti Kebangsaan Malaysia (UKM)

[2]Dr., Department of Information Science, Faculty of Science and Technology, Universiti Kebangsaan Malaysia (UKM)

E-mail: [1]ahmed.salih.2015@ieee.org , [2]ravie@ukm.edu.my

ABSTRACT

Infrastructure as a service (IaaS) is one of the most well-known cloud services. It facilitates high levels of flexibility in virtual machines (VMs). One of the challenges in cloud services is the effective management of large amounts of VM images. Cloud input-output (IO) performance significantly affects VMs, and significant storage resources are required, leading to higher management costs. Currently optimization is accomplished by two methods, involving either improving performance or decreasing image size, but low consumption of storage resources and achieving high levels of IO performance cannot be accomplished at the same time. Zone-based methods can potentially balance these requirements. The method proposed in this paper involves the use of adaptive replacement cache (ARC) and probabilistic content placement (PROB) algorithms, which together are known as zone based-adaptive replacement cache and probabilistic content placement (ZB-ARCPROB). This method provides more support to the cache management of images while considering all means of achieving high IO performance and low storage consumption. The research reported in this paper evaluated the ZB-ARCPROB method in relation to cloud performance using Network Simulator version 2.35 (NS2). The performance of the ZB-ARCPROB method was compared with zone-base method in terms of three metrics, namely IO latency, IO throughput, and relative storage consumption. This method is not only methodologically valid, but also offers the prospect of attracting interest from academia and industry. The results of the comparison indicate that the proposed ZB-ARCPROB method outperforms the zone-base method.

Keywords: *cloud computing, virtual machine (VM), image storage, adaptive replacement algorithm (ARC), probabilistic content placement*

1. INTRODUCTION

The primary aim of virtual technology is to replace physical with virtual resources. This reduces user costs and improves resource utilisation. Infrastructure as a Service (IaaS) is one of the most popular and widely used means of accomplishing this [4]. IaaS provides a customizable computing environment embedded in a virtual machine (VM) image. A VM runs as a complete and independent operation on a physical machine. Typically, as is the case in physical storage, the virtual disc stores the operating system and applications on the VM. Most images in IaaS clouds are set up as virtual discs that respond to cloud input-output (IO) requests when the VMs are operational, and are stored as ordinary files to maintain a complete computing environment while the VMs are not operating [6]. The major challenge is the effective management of data with the current rapid growth of VMs in clouds. IO performance is significantly affected by the organisation and storage of data in VMs. Service providers separate the cloud computing systems from storage systems, especially in large-scale clouds. VM performance tends to be compromised by different users loading the same image. While copy-write technology allows VMs to access local images, thereby reducing the load on network [7], it increases the prevalence of hot spots within the images [2].

© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Additionally, system status is significantly affected by management operations. Images comprise only one type of data, so their management is no different from managing any type of data in the cloud. Generally, VM creation and migration is similar to the distribution and migration of image files. A drawback associated with this form of management is its costliness in terms of bandwidth, and increasing file sizes leads to an increase in management overheads as larger images take longer to transfer. This means that many storage resources are wasted due to the presence of redundant files, which is unacceptable. Both service users and providers want to achieve better results depending on their requirements, including low management requirements, high storage utilisation and IO performance. The classical storage systems, however, cannot satisfy all of these requirements simultaneously. Most classical systems focus solely on IO performance [3], although some attempt to manage other requirements [8]. The main reason for focusing only on IO performance is due to the trade-offs that must be made between requirements. It is difficult, for example, to combine low management overheads with high IO performance, as the best way to improve IO performance is by caching the data, but saving data centrally is necessary if overheads are to be kept low. Zone-based methods partition computing nodes into many zones, and construct shared storage in each zone in order to cache hot data in the interests of high IO performance [1].

Another difficulty arises in that high levels of storage utilisation and high IO performance conflict. In order to increase storage utilisation, redundant data should be reduced as much as possible. The best way to achieve this is by storing the data centrally.

This paper proposes a new model for image storage that offers high IO performance without high storage consumption. The evaluations presented here show that the solution proposed in this paper improves IO performance to a greater extent than do other optimised solutions with similar management costs and storage consumption [11]. The contributions offered in this paper include a brief study and classification of some real-world models, examining their strengths and weaknesses, and analysis of the requirements associated with the management of VMs. This paper also proposes a new method that resolves some of the drawbacks that characterise other methods.

The remainder of this paper is organised as follows. Section two describes the background and related work. Section three sets out the research methodology, while section four presents the results and discussion. Finally, section five draws together the findings of this research in the conclusion.

2. BACKGROUND

This section is divided into two parts. The first part consists of a brief explanation of VM images. Secondly, the related work is discussed.

2.1 Virtual Machine Images

The main cloud center consists of managing and computing nodes, which share storage and network devices such as routers and switches. CPUs and memory are core resources which are provided by computing nodes. Multiple nodes can be created and operated on same cloud because of the resources that are used. VMs can be stored on local discs or remote discs on computing nodes. Fibre or Ethernet connections are used to connect the storage and computing nodes. There is more than one way of locating images, the means by which this is done being dependent on the model that is used. For local storage, there is no need for a network connection to transfer the image data. Otherwise, requests should redirect through switches and communication devices.

A critical aspect in this consideration is image format. Usually the VM generates a blank image with a specific format that is compatible with the operating system and the machine that is being used as a server. A drawback associated with this method is the time it takes; however, by using the features made available by the latest technology, this problem can be solved through the use of advance image formats. The copy on write (CoW) mechanism forms the basis for this approach [5]. Different image formats, for example JPEG and RAW, have different characteristics: RAW files possess flexibility and are characterised by their compatibility with other formats, but their file sizes are large compared to JPEG files. RAW images always constitute the base image format, but because of their large file sizes and many transfers occurring due to sharing, they typically consume resources and cause hot spots.

Images may be stored locally or remotely. Remote storage is typically used by large-scale data centers. Remote storage consists of multiple nodes, each of which is connected to the global center. In this type of cloud, the data is stored in shared storage, while the VMs are located in computing nodes. Because the VMs and image storage are highly dependent on each other and are stored in different ways, they are unable to work at high

© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

performance levels without specialist hardware, which currently is not available on the market. Global clouds that offer commercial services are typically loosely connected using Ethernet where, depending on network connections, VMs' performance may be compromised. The solution to this problem is to store the VMs on local discs. This, however, leads to other problems such as the need for more expensive network environments, which will in turn occupy valuable storage, and any malfunction of local discs will engender the destruction of the VMs. Much research has been done with the aim of solving these problems by exploring new models or new image formats. Until now, however, there is no standard approach that addresses all factors including storage utilisation and IO performance.

2.2 Related work

Much research continues to be done in this field. Some concerns the enhancement of image format, while others concentrate on the modification of the CoW mechanism. Researchers at IBM have proposed a new method for optimising image format called FVD [9]. Additionally, they have introduced a new mechanism called copy on read (CoR), which they have integrated with CoW. This integration of the two mechanisms has led to improvements in cache reading performance. Other research has sought to improve CoR in terms of efficiency, but this has the drawback that its use is limited to single machines, with the result that it is not suitable for large-scale applications [13]. Some researchers have attempted to solve this issue more generally [12]. Other work has concentrated on using new technology to distribute files according to image size, and then save them in various nodes of the VM, while Bogdan Nicolae has proposed a technique whereby distributed images are saved on local discs that operate as computing nodes [10]. The benefits delivered by this technique include the reduction of costs incurred by transmission and an increase in utilisation. Also, two mechanisms may be applied to decrease addressing costs, which are known as clone and shadow. The zone-based method that is proposed here aims to reduce costs as well as increasing network performance by enhancing the VM technique instead of reducing network latency. The zone-based model modifies the normal global-local storage by adding a cache tier to form a global storage facility, but using a random placement strategy to randomly schedule VMs' images to the computing nodes. A newly implemented technology has been chosen as a bench mark [1]. In the algorithm proposed in this paper, resource utilisation is increased by optimising the VM; an approach which this research suggests is a superior solution.

3. METHODOLOGY

This section describes the main process involved in developing image storage on clouds. The configuration of the proposed scenario along with the topology are introduced and discussed. This research contributes to the storage of virtual machine images in clouds by embedding the placement and replacement algorithms into the process of caching the image elements on clouds [1], and by proposing a ZB-ARC method to balance the storage of images in an efficient way by dividing the nodes into many zones together with consideration of the assigning of shared storage in each zone to cache data for high IO performance and low storage consumption. Figure 1 shows the zone-based model proposed by [1], which was used in this research as a benchmark against which the model proposed in this research may be evaluated. The model proposed in this research uses the placement algorithm constant probability of caching (PROB) and the replacement algorithm adaptive replacement cache (ARC) to provide additional support to the cache management of images.

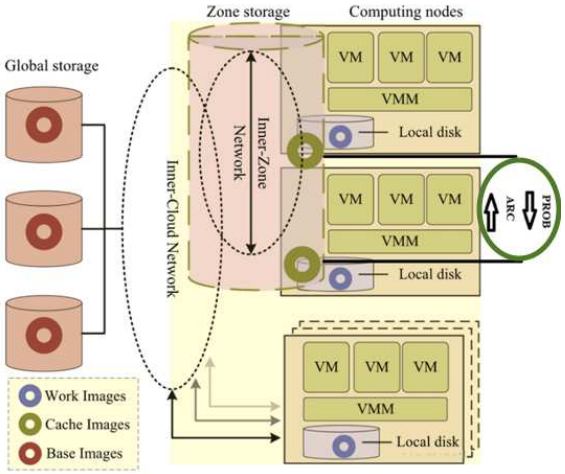

Figure 1: ZB-ARCPROB Model

3.1 Cache policy

The key concept of this research is aggressive cache replacement and placement policy. In a standard cache scenario, entries can be added to the cache when they are scheduled to run, based on the policy that describes what is to be cached. Similarly, when an image is destroyed it can be deleted from the cache. Cache memory restrictions may also cause a cache entry to be deleted if it has a

Journal of Theoretical and Applied Information Technology
29th February 2016. Vol.84. No.3
© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

lower priority than other entries waiting to be run. The point is that in standard cache management situations, adding items to the cache and deleting items from the cache are typically decoupled.

In the aggressive cache strategy, a new cache entry is added immediately after an existing one is deleted, provided that there is sufficient available memory for this to take place. It is not necessarily the case that the new cache entry is scheduled to be run immediately; however, by keeping the cache full at all times, the system can avoid the cost of fetching an entry newly scheduled to be run, because that entry will already be present in the cache in a paused state [5]. It is the placement algorithm (PROB) and the replacement algorithm (ARC) that facilitate this. The results of this approach show that this aggressive cache policy has the desired effect of improving performance in all three metrics simultaneously.

3.2 Replacement Algorithm (ARC) configuration

ARC replacement policy was used in this research in order to provide a direct sequential read for one-time-only. This includes processing the stored images such that they may pass through the cache without external events affecting the temporal locality. The ARC replacement algorithm was also assumed to manage long periods of VM low temporal locality by simultaneously keeping the VMs in the cache that are likely to be requested again by the server. With this in mind, it can be asserted that ARC replacement acts as a filter that is effective in distinguishing and tracking the temporal locality of VMs. In the event that sudden load in the network is imposed, ARC will then determine its source and reconfigure itself to exploit the opportunity [15]. In this research, it was assumed that the optimal conditions for operating the node may be based on the received request vector r at the maximum average request rate. If FC(r) presents the function that returns the largest values from r, C, then the cache probability vector for node k, Ck, is as shown in the following equation:

$$c_k^{LFU} = 1_{i \in F_C(r_k)} \quad (1)$$

where 1 represents the indicator function. In the case of the average VMs' request rates being in line with the ARC static, images with high average request rates in cache VMs are used instead.

3.3 Placement Algorithm (PROB) configuration

The probabilistic content placement algorithm was configured here by placing the VMs into the constant probability of caching used in identifying the origin of the client's request [14]. The local copy with probability p is preserved in the VMs instead of the intermediate cache and, as a result of this, the ARC algorithm is invoked. This is because the VMs' copy in this research was not positioned on the intermediate cache with a probability of 1-p.

4. EVALUATION

The topology used in this research is a tree-based network topology as shown in Figure 2. The topology consists of a number of links operating within a certain bandwidth. Here, a number of VMs (N) are assigned to a computing node. These nodes play a vital role in establishing communication between the server and router by issuing IO requests at a high speed.

For each node sent or received from the VMs in the course of communication, there is a consistent value represented by the node (v') and a local router (rv'). A random distribution of links was decided upon here in order to provide greater insight into the performance efficiency of the proposed model. In the example shown in Figure 2, each node has a local router through which the communication flow from node 1 to node n has a corresponding path: node 1, router R1, VM. Additionally, since the image may be distributed among separate locations, due consideration was given to the random placement of an image in the database.

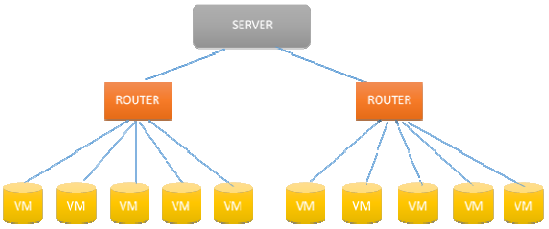

Figure 2: Tree-based network topology

As mentioned previously, three metrics relevant to academic and industrial concerns, namely IO latency, IO throughput and relative storage consumption, were used to evaluate the models considered here.

4.1 I/O latency

This research relied on the main network performance metrics used by [1] in order to ensure consistent results when comparing performance. It was assumed that one-way latency of requests transferring from VMs to other images can result in single or multiple switches. As such, the resultant switches that correspond with queues in the intersection of VMs and their images can lead to an

Journal of Theoretical and Applied Information Technology
29th February 2016. Vol.84. No.3
© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

increase in the VMs' total latency. As the main focus of this research is to observe the network performance when experiencing a delay due to long queues, the time transit of packets through Ethernet was considered. With this in mind, the latency of a switch (w) providing the necessary support for managing the received package with speed λ such as L(w)=f(a, λ) was considered. The total latency in this respect was calculated based on the amount of VM requests for the target image:

$$L = \sum_{w\ in\ path} L(w) \quad (2)$$

Here, path refers to the value of the link path between a VM and its image. This means that IO performance is likely to increase when the L decreases.

4.2 I/O throughput

Throughput was also used here to examine the performance of the proposed model compared to the zone-base method [1]. Throughput in this research refers to the total value of IO requests occurring in a certain time in one VM. The notion of having all switches implement effective policy for scheduling requests consists of the bandwidth of links when they are being shared by VMs. Accordingly, the throughput of a VM is either the sending rate of requests if the link is idle, or the actual bandwidth assigned to the VM if the link is congested. The eventual throughput of a VM can, therefore, be calculated by:

$$T = Min_{Eij \subseteq Epath} T(Eij) \quad (3)$$

4.3 Relative Storage consumption (RSC)

RSC represents the total ratio of the cloud storage image intake against a predefined baseline. In this research, the baseline was considered to be the original virtualised environment through which a certain VM corresponds to a number of images created when moving images from one node to another. The storage capability of the baseline can, therefore, be extended to accept additional images. This is essential to provide more space while utilising items from which unwanted data is removed. As such, certain values of VM may be of a lower value than one complete image. Hence, storage consumption in high-traffic networks can be changed based on the baseline value. In this research, RSC was used to compare the proposed model's storage consumption rates with those of [1] based on:

$$y = \frac{\sum_{i=1}^{n} space(Vi)}{\sum_{i=1}^{n} space(template)} \quad (4)$$

Here, *n* refers to the total value of VMs, while *space (Vi)* represents the image size of a VM, and *space (template)* refers to the template image size in the baseline case. [1] states that a smaller result in more free storage resources. If one method gets a smaller than the other methods, it occupies fewer storage resources to support the same amount VMs.

Table 1 shows the simulation parameters used in configuring the simulation environment. NS2 was used for the purpose of simulating the proposed model. The use of NS2 enabled implementation of the proposed model, within which it is possible to simultaneously improve each of the three metrics.

Table 1: Simulation parameters

Description	Value	Unit
No. of nodes	Variable	Node
Network area size	100	m²
Simulation time	30	Second
Node transmission range	30	m
Bandwidth	5	mbps

5. RESULTS AND DISCUSSION

Figure 3 shows the throughput values for the proposed model. The results show that ARC provides a high throughput value of 2.733, whereas the throughput value for the zone-base method is 2.610 as shown in Figure 4.

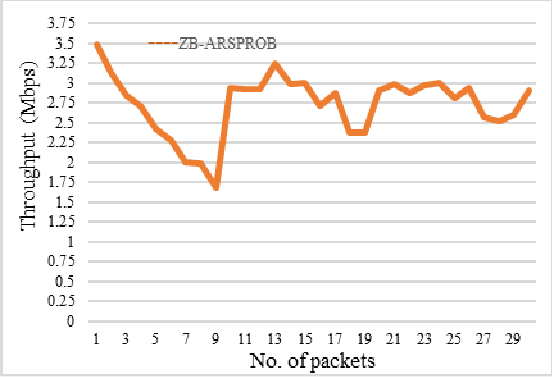

Figure 3: Throughput Values Of The Proposed ZB-ARCPROB

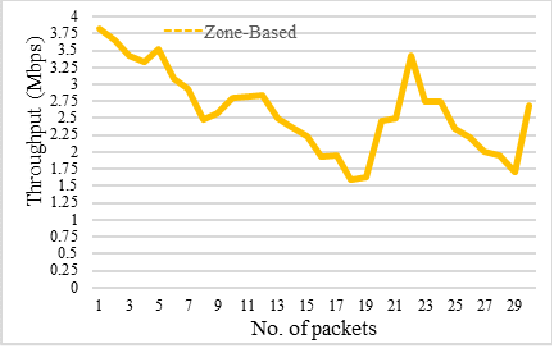

Figure 4: Throughput Values Of The Zone-Based Method

Journal of Theoretical and Applied Information Technology
29th February 2016. Vol.84. No.3
© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The comparison outcome shown in Figure 5 reveals that the proposed ZB-ARCPROB method delivers considerably higher throughput value than the zone-based method. This may be because the proposed model's replacement mechanism helps to organise and reference the images. This also assists in improving buffer management and page cache performance in the computed nodes. As such, it can be concluded that the proposed method throughput significantly exceeds that of the zone-based method, which consequently offers potential for improved image storage in cloud-related settings.

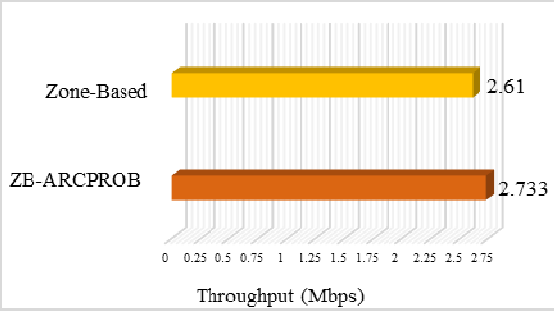

Figure 5: Throughput Comparison

The throughput performance of the proposed method compared to that of the zone-based method was also considered by examination of the number of nodes. This is necessary in order to ensure the stability and integrity of the proposed method. The results shown in Figure 6 reveal that the proposed method yields a hit rate higher than that of the zone-based method. It should be noted that the proposed model resulted in higher throughput values when communicating with between 10 nodes and 18 nodes.

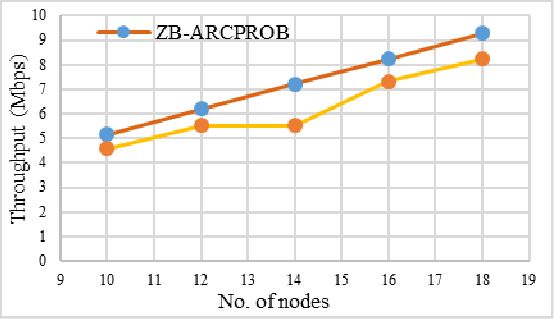

Figure 6: Throughput Differences With Regard To The Number Of Nodes

Figure 7 shows the latency value of the proposed method ZB-ARCPROB which yields a 0.002ms latency time, whereas the zone-based method resulted in latency time of 0.003ms as shown in Figure 8. From this, it can be concluded that the proposed model offers a lower latency time than the zone-based model. This may be due to the advantage of having images processed by the embedded algorithms.

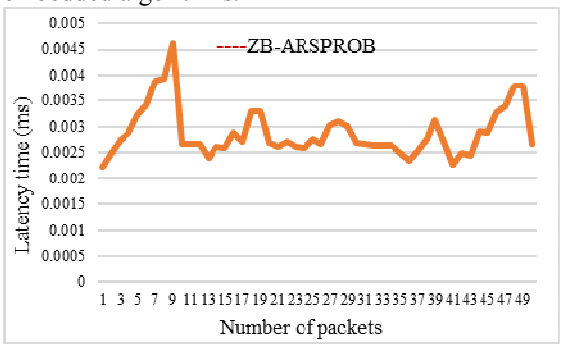

Figure 7: Latency Value of ZB-ARCPROB

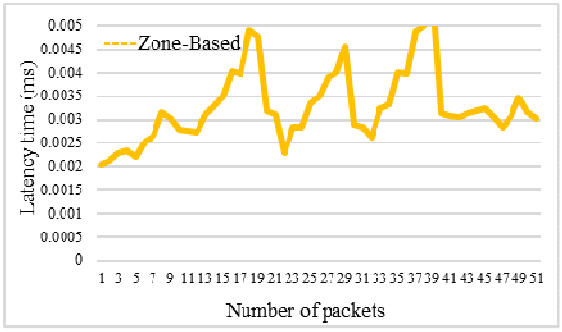

Figure 8: Latency Value Of The Zone-Base

The outcomes of from the latency results shown in Figure 9 reveal that the proposed model exhibits an IO latency of 30% less than that of the zone-based model. This indicates that the embedded cache algorithms used in this research provide the necessary antecedents for reducing remote requests when VMs require base image data.

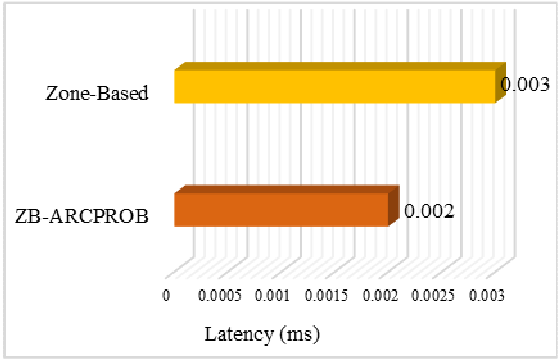

Figure 9: A Comparison Of Latency Value

Similar results were found when comparing the latency time results of the proposed method and zone-based method by the number of nodes. Figure 10 shows that the proposed ZB-ARCPROB model offers a stable and low latency rate. This is due to the placement and replacement algorithms boosting performance by planning cache storage in the VM.

Journal of Theoretical and Applied Information Technology
29th February 2016. Vol.84. No.3
© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

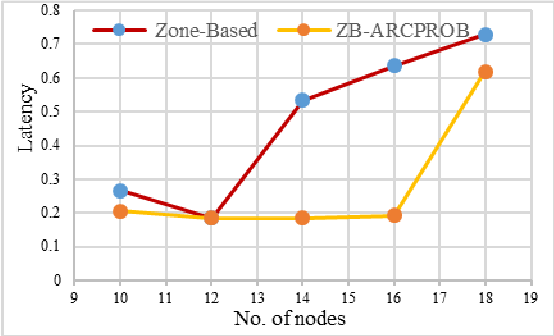

Figure 10: A Comparison Of Latency Value With Regards To The Number Of Nodes

Figure 11 shows the RSC of the proposed model in different storage scenarios. The results obtained showed that, in three storage scenarios, the proposed method delivers lower RSC values of 252.84 (storage 1), 269.72 (storage 2) and 246.57 (storage 3). The zone-based method, however, delivers higher RSC values of 363.67 (storage 1), 373.11 (storage 2) and 379.81 (storage 3). This is an interesting finding in which the proposed model proved to offer lower RSC values in all of the storage scenarios.

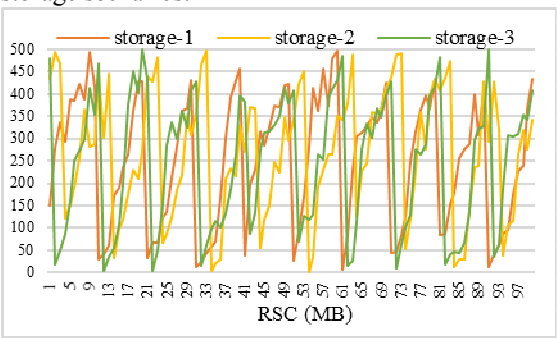

Figure 11: RSC of ZB-ARCPROB

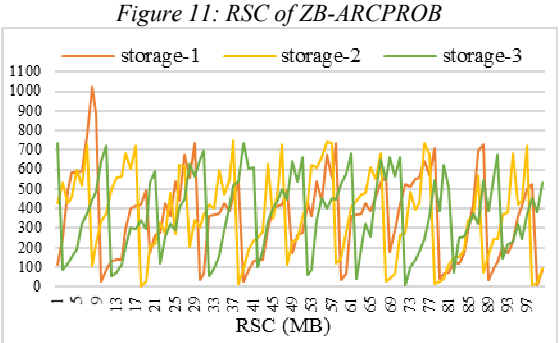

Figure 12: RSC Of The Zone-Based Method

Figure 13 shows that the proposed method yields lower RSC values than the zone-based method. This may be due to the zone-base method adding storage consumption to some degree into the cache tier. It should be noted that the zone-base method involves sharing in a zone, while the proposed method is better organised around the zone, thereby generating less redundant data. Thus, the proposed method consumes fewer storage resources than the zone-based method.

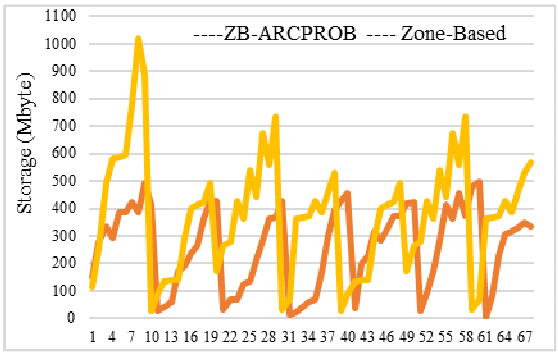

Figure 13: RSC Comparison

The results show that the proposed method (ZB-ARCPROB) offers higher throughput values, lower latency time, and lower storage consumption than the zone-based method. These are interesting findings which suggest that using proper cache management algorithms may provide remarkable improvements in performance which dramatically influence the distribution of nodes in VMs. The use of these algorithms maximizes throughput between application nodes and storage nodes, which as a result delivers lower latency values and reduces storage requirements.

6. CONCLUSION

The increasing popularity of cloud computing creates a huge and growing number of virtual machine images, more of which are stored in data centers than ever before. The efficient management of these images represents a big challenge to cloud managers. Ideally, users expect a perfect image storage system that is capable of providing high IO performance and high resource utilisation. These requirements, however, are hard to meet simultaneously. A feasible solution is to trade off these requirements, managing images flexibly in different scenarios. This paper has analysed the root causes of this problem in detail and discussed the advantages and drawbacks of some traditional image storage models. This paper has also presented key metrics by which the effectiveness of managing images may be evaluated. In order to satisfy the above requirements as far as possible, this paper proposes a new image storage method, analyses its characteristics and compares it with other typical methods. The results of this evaluation show that the proposed method offers substantial improvements in IO performance without adding

© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

extra management costs and storage consumption, and has simultaneous scalability and availability.

ACKNOWLEDGEMENT

This study was supported by Dana Impak Perdana (DIP-2014-037) of the Universiti Kebangsaan Malaysia (UKM).

REFERENCES:

[1] X. Xu, H. Jin, S. Wu, and Y. Wang, "Rethink the storage of virtual machine images in clouds," *Futur. Gener. Comput. Syst.*, 2015.

[2] Shaw, S. B., Singh, A. K., "Use of proactive and reactive hotspot detection technique to reduce the number of virtual machine migration and energy consumption in cloud data center", *Computers & Electrical Engineering*, 2015.

[3] Ibrahim Abdullahi, Suki Arif, Suhaidi Hassan, "Survey on caching approaches in Information Centric Networking", *Journal of Network and Computer Applications 56*, 2015, 48–59.

[4] S. S. Manvi, G. Krishna Shyam, "Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey," *J. Netw. Comput. Appl.*, vol. 41, no. 1, pp. 424–440, 2014.

[5] Chen, J. Wang, J. Tan, Z. Xie C., "Recursive Updates in Copy-on-write File Systems-Modeling and Analysis", *Journal of Computers,* 2014, 9(10), 2342-2351.

[6] V. Tarasov, D. Hildebrand, G. Kuenning, and E. Zadok, "Virtual Machine Workloads: The Case for New Benchmarks for NAS 1," *Fast '13*, 2013, pp. 307–320.

[7] S. Bazarbayev, M. Hiltunen, K. Joshi, W. H. Sanders, and R. Schlichting, "Content-based scheduling of virtual machines (VMs) in the cloud," *Proc. - Int. Conf. Distrib. Comput. Syst.*, 2013, pp. 93–101.

[8] S.M. Shamsheer Daula, Dr. K.E Sreenivasa Murthy, G Amjad Khan, "A Throughput Analysis on Page Replacement Algorithms in Cache Memory Management", *International Journal of Engineering Research and Applications (IJERA),* Vol. 2, Issue 2, Mar-Apr 2012, pp.126-130.

[9] C. Tang, "Fvd: a high-performance virtual machine image format for cloud," *Proc. 2011 USENIX Conf. USENIX Annu. Tech. Conf.*, 2011, pp. 18–24.

[10] B. Nicolae, J. Bresnahan, K. Keahey, and G. Antoniu, "Going Back and Forth: Efficient Multideployment and Multisnapshotting on Clouds," *Hpdc 11 Proc. 20th Int. Symp. High Perform. Distrib. Comput.* , 2011, pp. 147–158,

[11] Amit S. Chavan, Kartik R. Nayak, Keval D. Vora, Manish D. Purohit and Pramila M. Chawan, "A Comparison of Page Replacement Algorithms", *IACSIT International Journal of Engineering and Technology,* Vol.3, No.2, April 2011.

[12] D. T. Meyer, G. Aggarwal, B. Cully, G. Lefebvre, M. J. Feeley, N. C. Hutchinson, and A. Warfield, "Parallax: virtual disks for virtual machines," *in Proceedings of the 3rd ACM SIGOPS / EuroSys European Conference on Computer Systems - Eurosys'08*, 2008, p. 41.

[13] R. Chandra, N. Zeldovich, C. Sapuntzakis, and M. S. Lam, "The Collective: A Cache-Based System Management Architecture," *Proc. 2nd Conf. Symp. Networked Syst. Des. Implementation-Volume 2*, pp. 259–272, 2005.

[14] Jelenkovic, P. R., & Radovanovic, "A Optimizing LRU caching for variable document sizes, Combinatorics, Probability and Computing", 2004, 13(4-5), 627-643.

[15] Nimrod Megiddo and Dharmendra S. Modha, "ARC: A Self-Tuning, Low Overhead Replacement Cache.," *FAST'03 Proc. 2nd* USENIX Conf. File storage Technol., vol. 3, pp. 115–130, 2003.

© 2005 - 2016 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

APPENDIX

INPUT: The requested page *x*
INITIALIZATION: Set *p* = 0 and set lists *T*1, *B*1, *T*2, and *B*2 to empty
1: **if** (*x* is in *T*1 fi *T*2) **then** *Û* cache hit
2: Move *x* to the top of *T*2.
3: **else if** (*x* is in *B*1) **then**
4: Adaptation: Update *p* = min{*p* + max{1, |*B*2|/|*B*1|},*N*} 5: Replace()
6: Move *x* to the top of *T*2 and place it in cache.
7: **else if** (*x* is in *B*2) **then**
8: Adaptation: Update: *p* = max{*p* ≠ max{1, |*B*1|/|*B*2|}, 0} 9: Replace()
10: Move *x* to the top of *T*2 and place it in cache.
11: **else** *Û* cache and directory miss
12: **if** (|*T*1| + |*B*1| = *N*) **then**
13: **if** (|*T*1| < *N*) **then**
14: Discard LRU page in *B*1.
15: Replace()
16: **else**
17: Discard LRU page in *T*1 and remove it from cache.
18: **end if**
19: **else if** ((|*T*1| + |*B*1| < *N*) and (|*T*1| + |*T*2| + |*B*1| + |*B*2| O *N*)) **then**
20: **if** (|*T*1| + |*T*2| + |*B*1| + |*B*2| = 2*N*) **then**
21: Discard LRU page in *B*2.
22: **end if**
23: Replace()
24: **end if**
25: **end if**
Replace()
Set *Prob* =0.4
Generate a random number *r* [0, 1]
26: **if** ((|*T*1| >=1) and ((*x* is in *B*2 and |*T*1| = *p*) or (|*T*1| > *p*))) or (*r*>=*Prob*) **then**
27: Delete LRU page in *T*1 (also remove it from cache); move it to MRU position in *B*1.
28: **else**
29: Delete LRU page in *T*2 (also remove it from cache); move it to MRU position in *B*2.
30: **end if**