Set Cooperative Cache for Virtual Machine Relocation

Cong Hu
Anhui Electric Power Corporation, Information & Telecommunication Branch
Hefei, China
E-mail: huc0019@163.com

Abstract—With cloud computing models gaining significant momentum, data centers are increasingly employing virtualization as a means to support a large number of heterogeneous workloads running simultaneously on a multicore server. In such environments, contention for shared cache space can have a destructive effect on performance isolation among virtual machines. However, the existing cache design usually ignores the variety of tenant's service requirements and the inherent characteristics of virtual machines, such as VM relocation, which means the change of vCPU-to-core mapping. In this paper, we propose a set cooperative cache to optimize the access latency of relocated VM in the cache bank level, which tries to retain the evicted data of high pressure cache sets in corresponding low pressure sets. In a simulated 16 core system, the set cooperative cache can reduce the cache miss rate by 25.2% on average, and improve the IPC performance by 4.2% compared to the traditional LRU cache policy.

Keywords-Cloud Computing; Relocation; Virtualization; VM Relocation; Cache

I. INTRODUCTION

Since the emergence of cloud computing[1,2] through the continuous development of science and technology, through the advancement of academia and industry, the application of cloud computing is developing continuously and deeply, cloud computing is also from theory to practice. With cloud computing technology matures, data center development[3,4]. Today's data center is not only a simple server hosting, maintenance, it has become a collection of large amount of data operation and storage as one of the high performance computer centre.

Cloud computing data center involving the large number of servers, the servers usually use multi-core processors. Multi-core processors have become the only way to technology for modern high performance microprocessor structure, it through in a single chip processor core is put in the multiple structure is relatively simple and overcome the traditional processor design in such aspects as performance, power consumption, heat dissipation, verify the problem[5-14].

In a virtualization system, virtual CPUs (vCPUs) and physical CPUs (pCPUs) mapping is not fixed. Hypervisor vCPUs relocation to different pCPUs improves the efficiency of the use of physical processors as possible. Hypervisor usually schedules the strategy according to its relocation decision. Xen default scheduler based on credit, this is a can guarantee the global load balancing multi-core system of proportional sharing scheduler. The scheduler in each scheduling cycle for each vCPU allocates a certain amount of time, called credit. When running, vCPUs can consume the assigned time slice. In order to guarantee the fairness, the scheduler is always scheduling those who still have remaining credit vCPU run 30 ms time slice.

In a multi-core system, in order to ensure the load balancing, credit scheduler always waits for dynamically vCPUs relocation to idle CPU cores. When a physical processor core of all vCPUs spent their time slice, the scheduler will steal from the other busy core a waiting vCPU, there are still remaining credit and assigned to spare cores. This default layer scheduling policy did not consider cost brought by the migration, is radically migration between the physical processor cores, and makes the nuclear busy as far as possible. Experimental results show that the typical load relocation cycle to an average of 178.1 ms, and the worst cases, only 0.1 ms.

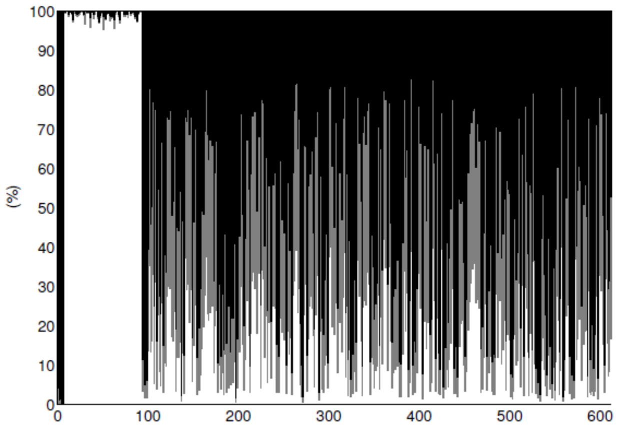


Figure 1. The distribution of 473.astar group cache access, high pressure (black), pressure (gray), low pressure in the (white). Once every 107k visit sampling.

In fact, the load of the working set in cache between groups (Set) was usually not uniformly distributed. In Fig. 1 shows the SPEC CPU2006 astar program in the execution of each Cache the distribution of group visit, it is assumed that a capacity of 2 MB, block size of 64 bytes of 8-way set associative Cache. Experiment, each cache group use a maximum of 15 saturated counters to record the group's visit, when accessing the group loss occurs, the value of the counter plus 1, minus 1 conversely. According to the value of the counter, cache group access pressure can be divided into low (0 to 5), (6-10) and high (11-15) three types. As shown in Fig. 1, when the initialization phase is completed, the cache of each group to visit pressure distribution is extremely uneven.

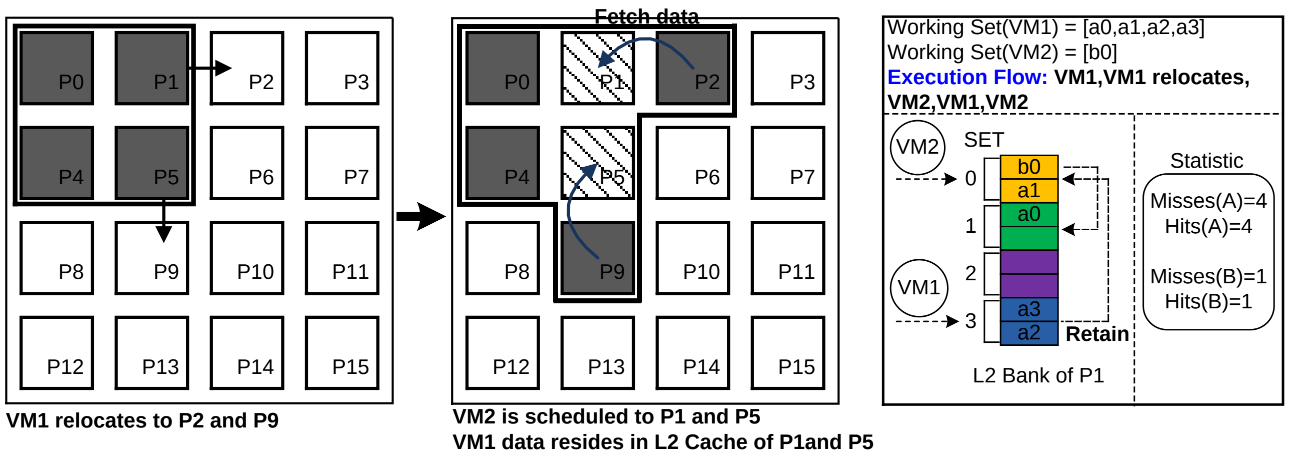

P0
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
P13
P14
P15
VM1 relocates to P2 and P9
Fetch data
VM2 is scheduled to P1 and P5
VM1 data resides in L2 Cache of P1and P5
Working Set(VM1) = [a0,a1,a2,a3]
Working Set(VM2) = [b0]
Execution Flow: VM1,VM1 relocates, VM2,VM1,VM2
VM2
VM1
SET
0
1
2
3
b0
a1
a0
a3
a2
Retain
L2 Bank of P1
Statistic
Misses(A)=4
Hits(A)=4
Misses(B)=1
Hits(B)=1

Figure 2. Basic idea of cache set co-operation mechanism

II. CACHE SET CO-OPERATION MECHANISM

A. Basic Idea

In the current virtualization technology, vCPU to pCPU mapping is not fixed. Virtual machine management program through the vCPU relocates to a different processor cores in order to improve the physical processor utilization. However, for the private Cache by relocation of the virtual machine data set is still retained in the original processor cores, when the original processor cores have the new virtual machine scheduling, these data may be eliminated because of conflict failure, leading to the relocation of the virtual machine's delay increased significantly. Typical load to fetch address was usually not uniformly distributed in the cache groups, when some high failure rate of the other group may still is in a state of underused. In this paper, we proposed set cooperative mechanism (SC), this mechanism support the Cache many-to-many Shared between groups, from the high capacity pressure group heading out of the virtual machine before the relocation of the working Set (source) in the flexibility to keep in the same Cache (target) of low utilization rate of group, thus extending the lifetime of the data, thus in the virtual machine access after the relocation to a higher hit ratio in order to improve Cache performance. SC mechanism should meet the following goals: (1), any point in the process of program execution should be allowed to relocation of the virtual machine data efficiently retained in any candidate Cache in the group; (2), should be flexible enough to attain the goal of a set of Shared by multiple source group (one-to-many) and multiple objective group was Shared by a source group (to a). As shown in Fig. 2, after VM1 relocation to P2 and P9, VM2 is scheduling to P1 and P5 execution at the same time, under the mechanism of SC, required data is retained in the P1 P2 SET0, characters, and in SET3, this greatly reduces the data access latency.

Specific, group collaboration between the Caches contains the following three components:

(1) Collaboration conditions: how to determine the source and destination Cache group. During load operation using a saturated counter analyzes the capacity of the pressure in each group, with statistical results formulate relevant indicators to identify the source and destination;

(2) Collaboration strategy: how to will be in the form of many-to-many sources in the group working set to retain in the target group. Needed in the relocation of data from the source virtual machine group to be eliminated when determined to keep the purpose of the group address;

(3) Search strategy: how to find the goal after the virtual machine relocation in the group retains the data block. When the virtual machine relocation to read data from the original Cache body, and determine the search order of multiple objective group, port competition and power consumption and other relevant factors to take into consideration here.

B. Collaboration Conditions

SC mechanism is the essence of the elimination from the high capacity pressure groups of data stored in the low utilization rate in the group, this paper use the failure of Cache groups within a certain time interval to measure the pressure, total pressure information can be in each of the Cache controller increases pressure a hardware form to statistical information, each corresponding to the form of a Cache group, and the use of Cache groups address to index the record. When lacking every visit, the corresponding record will be updated. And made the high pressure and low pressure threshold two indicators:

$$LPT = \min + \alpha \times (\max - \min)$$

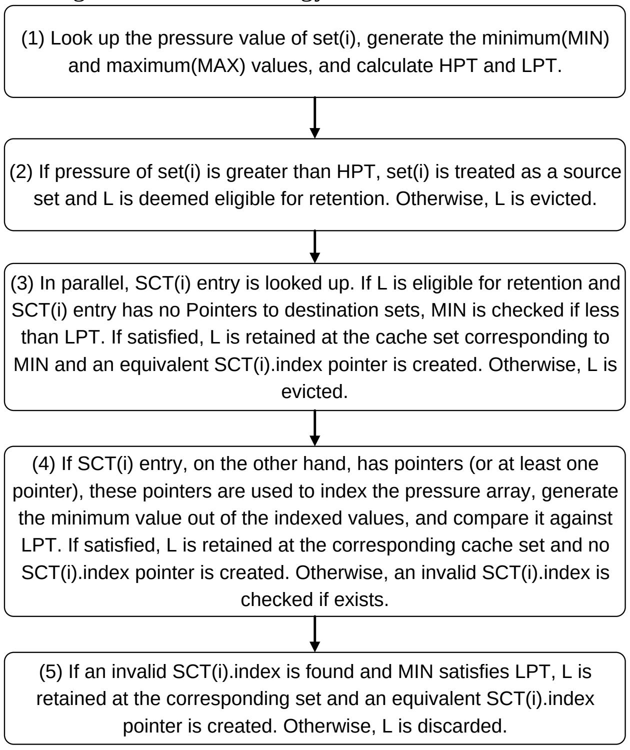
$$HPT = \max - \alpha \times (\max - \min)$$

The max and min record the number of failure maximum and minimum values, if a Cache set pressure is less than the LPT, then the group can be used as an objective group and receive any source of data; If a Cache group pressure is greater than HPT, this group is regarded as the source group, its data can be retained in multiple objective group. Can show you how to use the two thresholds can be achieved more group Shared, can realize the source and destination by adjusting the parameter set range of expansion and contraction. In this article the subsequent experiments, α value is 0.2.

C. *Collaboration Strategy*

Set cooperative need maintain a set of working Table (SCT) in each of the L2 Cache body. Each Cache in SCT has a corresponding set of tables. It is said that SCT item number is the same as the number of the L2 groups in the body. The first s Table in SCT item (s), hold at most K pointer, each valid pointer pointing to a purpose with different index group. Group collaboration between Cache can use the pointer location Cache block is retained.

When it needs to be eliminated in the first group I LRU piece L, collaboration strategy as shown in Fig. 3 are working collaboration strategy.



(1) Look up the pressure value of set(i), generate the minimum(MIN) and maximum(MAX) values, and calculate HPT and LPT.

(2) If pressure of set(i) is greater than HPT, set(i) is treated as a source set and L is deemed eligible for retention. Otherwise, L is evicted.

(3) In parallel, SCT(i) entry is looked up. If L is eligible for retention and SCT(i) entry has no Pointers to destination sets, MIN is checked if less than LPT. If satisfied, L is retained at the cache set corresponding to MIN and an equivalent SCT(i).index pointer is created. Otherwise, L is evicted.

(4) If SCT(i) entry, on the other hand, has pointers (or at least one pointer), these pointers are used to index the pressure array, generate the minimum value out of the indexed values, and compare it against LPT. If satisfied, L is retained at the corresponding cache set and no SCT(i).index pointer is created. Otherwise, an invalid SCT(i).index is checked if exists.

(5) If an invalid SCT(i).index is found and MIN satisfies LPT, L is retained at the corresponding set and an equivalent SCT(i).index pointer is created. Otherwise, L is discarded.

Figure 3. Collaboration Strategy

D. *Search Strategy*

When a request to access the Cache block B, the Cache controller concurrent search determined according to index of the B group s and group collaboration Table SCT (s). If it occurred in the group s accuracy can satisfy the request. Otherwise, SCT (s) identified in the Cache group need to be in order to find, until finally the second hit or all missing. Order to find the simplified group collaboration between the designs of the Cache, avoiding port competition and reduces the power consumption. When there is no second hit occurs, the group s pressure record will be updated and trigger the collaboration mechanism at the same time. At the same time read it will be in parallel the requested from the memory Cache block and inserted into the groups.

III. EXPERIMENTAL

The experiment is based on the open source system x86 emulators FeS_2 [15] as support multicore processor simulation platform virtualization technology. The simulator adopts precise execution drive the clock model, including the cache level, branch predictor and superscalar out-of-order processor core simulation, provides detailed and flexible multiprocessor storage model of the system clock. By changing the storage system in a simulator Ruby module implements the Cache collaboration mechanism between groups, in this paper, the simulation of processor cores configuration as shown in Table 1. Based on this processor core structure, this paper built a more than 16 nuclear simulation platform.

TABLE I. SIMULATION PLATFORM CONFIGURATION

Processor	frequency	4G
	Prefetch/launch	6/4
	Command window	80
	ROB	152
	Integer/float registers	104/80
	Integer FU	3 ALU
	float FU	2 ALU
L1 Cache	L1-iCache & L1-dCache	32kB/8way/64B/LRU
	port	2i / 2 d
	latency	4 Cycles
	L1 MSHRs	4 I / 32 d
L2 Cache	L2-Cache	2MB/8way/64B/LRU(+SC)
	port	1
	latency	14 Cycles
	L2 MSHR	32

This article use the open source Xen cloud platform to provide enterprise server virtualization support XCP, XCP contains can support Windows ® and a series of guest operating system such as Linux ® Xen Hypervisor, through the Open vSwitch technology achieved rich virtual network support, and provides support for cloud storage infrastructure; At the same time, its internal XAPI or XenAPI is a management protocol stack, is used to configure and control the Xen can make host and resource pool, and coordinates in the pool resources. XCP will server load together, can significantly reduce power consumption, save cooling and management expenses, suitable for sustainable computing environment.

Load performance testing program choice PARSEC [16] multithreaded program sets, it is recognized as a system structure research field for multi-core processor design multithreaded test program set, its application has a good parallelism. The program name and characteristics are shown in Table 2.

TABLE II. LOAD THE PROGRAM NAME AND MISS OF RUN SEPARATELY UNDER 2 MB CACHE SIZE

NAME	Cache Miss %	NAME	Cache Miss %
blackscholes	0.1	fluidanimate	0.4
bodytrack	2.1	freqmine	0.16
dedup	0.25	streamcluster	3.0
facesim	0.9	swaptions	0.01
ferret	1.5	vips -	0.14
x264	0.36	canneal	5.2

In the process of experiment, this article on the above 16 nuclear FeS_2 simulator successfully installed XCP, and can be run at the same time four installation debian 6.0 operating system virtual machine, four virtual machines running on the same the same benchmark.

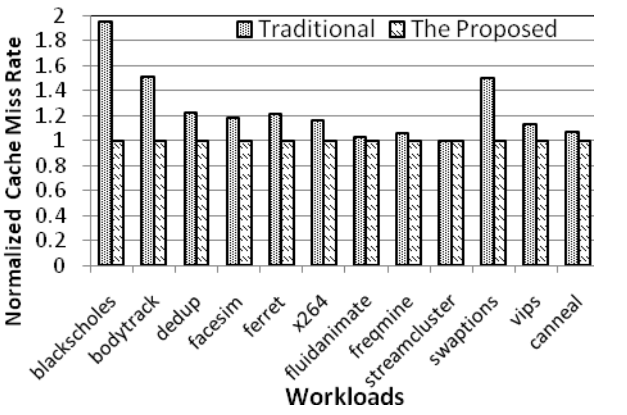


Figure 4. Missing rate of different load in different Cache Cache strategy

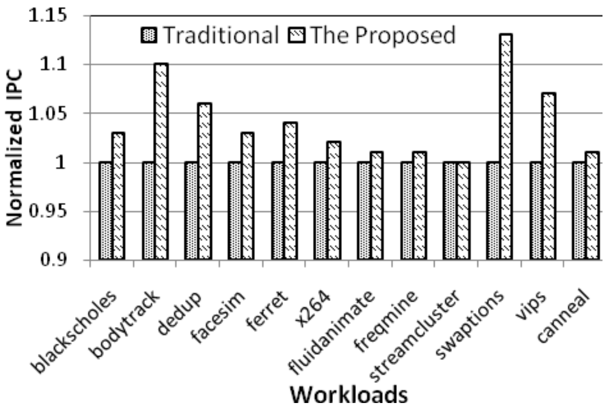


Figure 5. Performance of different load in different Cache Cache strategy

IV. RESULTS

This section put forward strategies of collaboration between the Cache groups in terms of performance assessment, indicators include the IPC and Cache missing rate. This section also extensibility of collaboration strategies were tested, in the same hardware configuration under the condition of increasing number of VMS, evaluating the performance of the average each VM change trend.

A. Performance Evaluation

As shown in Fig. 4, to Cache missing rate, this paper proposes the Cache mechanism of collaboration between groups relative to the traditional Cache strategy to reduce the average 25.2% less. Cache mechanism of collaboration between groups can have a big Cache performance improvement of reason mainly has two aspects: one is the coordination mechanism as far as possible keep the former data of a VM, so that when the original VM data access (without memory, greatly reduce the data access time. From the point of view, the capacity of this way of group collaboration between implicit added to the original VM Cache capacity, is bound to make the original VM Cache missing drop. The second is the imbalance between the coordination mechanism using the Cache feature, can not affect the current VM Cache characteristics and performance under the condition of full use of the capacity of the Cache.

As shown in Fig. 5, the mechanism of collaboration between Cache group IPC evaluation. Can be seen from the diagram, relative to traditional Cache strategy, this paper proposes the Cache mechanism of collaboration between groups on the performance of an average increase of 4.2%. Application performance is mainly because under the collaborative mechanism between groups, caused by their Cache missing rate decreases. In addition, combined with Fig. 4 can be found, for missing Cache down a big program is not IPC ascend the highest, that is because different programs have different behavior characteristics, Cache missing rate can affect the performance of the program, but is not a one-to-one correspondence between their linear relationship.

B. Scalability Ttesting

Fig. 6 is the results of proposed mechanism of collaboration between Cache group scalability evaluations. Can be found from the figure, with the rising number of VMS, average VM performance in declining, but compared with the traditional Cache, group collaboration mechanism performance degradation speed is slow, obviously in 6 VM, 21.1% higher performance than conventional Cache. Extensibility is superior to the traditional Cache mechanism of collaboration between the reasons is that group can more fully dig up the heat in the Cache data, improve the utilization rate of Cache capacity. In addition, in Fig. 6 shows that the mechanism of collaboration between both groups or traditional Cache strategy after more than 4 VM performance declines are big, this is because each VM is configured with four cpus, when the VM number greater than 4 indicates the physical hardware platform of the 16 core processing all the VM cannot run at the same time, which can cause the CPU time-sharing multiplexing and plenty of VM switch and relocation.

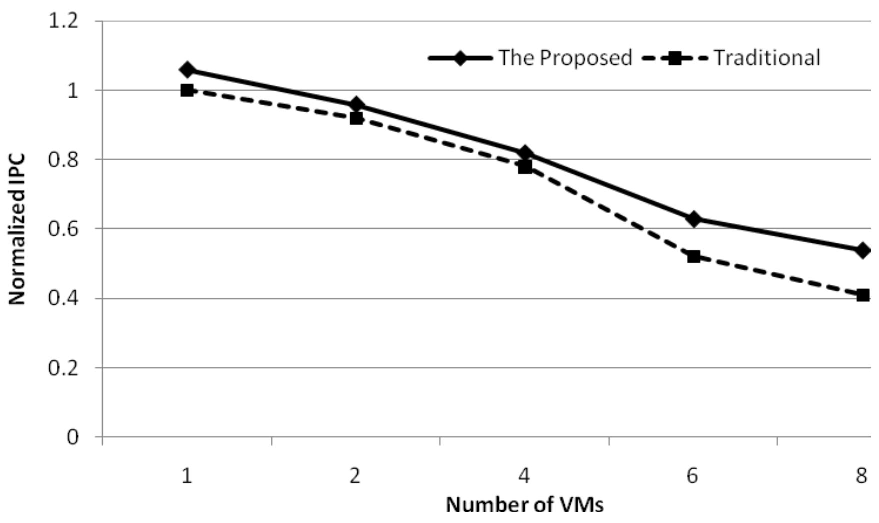


Figure 6. For vips load, different Cache strategies vary with the number of VM performance

V. CONCLUSIONS

Frequent reset bring serious performance degradation in a virtual scenario, in order to overcome this problem, this paper proposes a Cache mechanism of collaboration between groups. The mechanism of using Cache access imbalance features between groups, and have migrated off the virtual machine Cache data in the Cache remains in the original as possible use, so that we can effectively improve the utilization rate of Cache and reduce the loss of Cache rate when the data access. This paper proposes Cache mechanism both in performance or scalability, and the relatively traditional Cache strategy has obvious advantages.

REFERENCES

[1] Tambe, A., Trends and directions in networking - impact of virtualization and cloud, Proceedings of International Symposium on VLSI Technology, Systems and Application (VLSI-TSA), vol.1, no.1, pp.28-30, April 2014.

[2] Amazon Elastic Compute Cloud. http://aws.amazon.com/ec2.

[3] Jacob Leverich and Christos Kozyrakis. Reconciling high server utilizationand sub-millisecond quality-of-service. In Proceedings of the2014 EuroSys Conference, Amsterdam, Nethelands, 2014.

[4] Christina Delimitrou and Christos Kozyrakis. Quasar: Resourceefficient and QoS-aware cluster management. In Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'14, page 127–144, New York, NY, USA, 2014. ACM.

[5] Chen, Gang; Hu, Biao; Huang, Kai; Knoll, Alois; Huang, Kai; Liu, Di; Shared L2 Cache Management in Multicore Real-Time System, IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), vol.170, no.170, pp.11-13 May 2014

[6] MazenKharbutli and Rami Sheikh. LACS: a locality-awarecost-sensitive cache replacement algorithm. IEEETransactions on Computers, 2013, 6(3): 1-29.

[7] Babu, S.A.; Hareesh, M.J.; Martin, J.P.; Cherian, S.; Sastri, Y., System Performance Evaluation of Para Virtualization, Container Virtualization, and Full Virtualization Using Xen, OpenVZ, and XenServer,Fourth International Conference onAdvances in Computing and Communications (ICACC), vol.247, no.250, pp. 27-29 Aug. 2014.

[8] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In Proceedings of the ACM Symposium on Operating Systems Principles (SOSP), 2003.

[9] Michael Ferdman, AlmutazAdileh, OnurKocberber, Stavros Volos, Mohammad Alisafaee,DjordjeJevdjic, CansuKaynak, Adrian Daniel Popescu, Anastasia Ailamaki, and BabakFalsafi. Clearing the Clouds: A Study of Emerging Scale-out Workloads on Modern Hardware. In the 17th InternationalConference on Architectural Support for Programming Languages and Operating Systems, March 2012.

[10] Jaideep Moses, Ravi Iyer, Ramesh Illikkal, Sadagopan Srinivasan, Konstantinos Aisopos: Shared Resource Monitoring and Throughput Optimization in Cloud-Computing Datacenters. In Proceedings of the 2011 IEEE International Parallel and Distributed Processing Symposium: 1024-1033.

[11] Ravi Iyer, Ramesh Illikkal, Li Zhao, Don Newell, Jaideep Moses. Virtual platform architectures for resource metering in datacenters. In: ACM SIGMETRICS Performance Evaluation Review, Volume 37, Issue 2 (September 2009), pages: 89-90.

[12] D. Kim, H. Kim, and J. Huh. Virtual Snooping: Filtering Snoops inVirtualized Multi-cores, In Proceedings of International Symposium onMicroarchitecture, 2010.

[13] HarshadKasture and Daniel Sanchez. Ubik: Efficient cache sharing with strict qos for latency-critical workloads. In Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS '14, page 729–742, New York, NY, USA, 2014. ACM.

[14] N. Neelakantam, C. Blundell, J. Devietti, M. M. K. Martin,and C. Zilles. FeS2: A full-system execution-driven simulatorfor x86. In Poster session at ASPLOS '08, 2008. URLhttp://fes2.cs.uiuc.edu/acknowledgements.html.

[16] C. Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC benchmarksuite: Characterization and architectural implications. In Proceedings of PACT, October 2008.