User-Centric Heterogeneity-Aware MapReduce Job Provisioning in the Public Cloud

Eric Pettijohn and Yanfei Guo, *University of Colorado, Colorado Springs;* Palden Lama, *University of Texas at San Antonio;* Xiaobo Zhou, *University of Colorado, Colorado Springs*

https://www.usenix.org/conference/icac14/technical-sessions/presentation/pettijohn

This paper is included in the Proceedings of the
11th International Conference on Autonomic Computing (ICAC '14).
June 18–20, 2014 • Philadelphia, PA

ISBN 978-1-931971-11-9

Open access to the Proceedings of the
11th International Conference on
Autonomic Computing (ICAC '14)
is sponsored by USENIX.

User-Centric Heterogeneity-Aware MapReduce Job Provisioning in the Public Cloud

Eric Pettijohn*, Yanfei Guo*, Palden Lama† and Xiaobo Zhou*

*Department of Computer Science, University of Colorado, Colorado Springs, USA
†Department of Computer Science, University of Texas, San Antonio, USA
Emails: {epettij4, yguo, xzhou}@uccs.edu, palden.lama@utsa.edu

Abstract

Cloud datacenters are becoming increasingly heterogeneous with respect to the hardware on which virtual machine (VM) instances are hosted. As a result, ostensibly identical instances in the cloud show significant performance variability depending on the physical machines that host them. In our case study on Amazon's EC2 public cloud, we observe that the average execution time of Hadoop MapReduce jobs vary by up to 30% in spite of using identical VM instances for the Hadoop cluster. In this paper, we propose and develop U-CHAMPION, a user-centric middleware that automates job provisioning and configuration of the Hadoop MapReduce framework in a public cloud to improve job performance and reduce the cost of leasing VM instances. It addresses the unique challenges of hardware heterogeneity-aware job provisioning in the public cloud through a novel selective-instance-reacquisition technique. It applies a collaborative filtering technique based on UV Decomposition for online estimation of ad-hoc job execution time. We have implemented U-CHAMPION on Amazon EC2 and compared it with a representative automated MapReduce job provisioning system. Experimental results with the PUMA benchmarks show that U-CHAMPION improves MapReduce job performance and reduces the cost of leasing VM instances by as much as 21%.

1 Introduction

Today, big data processing frameworks such as Hadoop MapReduce [1] are increasingly deployed in public clouds. However, due to the absence of automation tools, currently end users are forced to make job provisioning decisions manually. Recent studies [16, 28, 29, 31] have focused on improving Hadoop job performance through automated resource allocation and parameter configuration. However, most research has been done on small private clusters, which tend to be homogeneous with respect to the hardware configuration and performance.

One of the foremost challenges of MapReduce job provisioning in a public cloud is imposed by the heterogeneity of the underlying hardware infrastructure [10,21, 27]. Cloud datacenters usually upgrade their hardware infrastructure over time, resulting in multiple generations of hardware with widely varying performance [25]. Such hardware heterogeneity has a significant impact on Hadoop job completion time [21]. However, the VM instances offered by public cloud providers do not indicate the performance implications of the heterogenous hardware that hosts them. Furthermore, there is no guarantee that one VM will always be provisioned on the same type of hardware. Our motivational case study on Amazon EC2 public cloud shows that the average execution time of Hadoop MapReduce jobs varies by up to 30% despite using identical VM instances for the Hadoop cluster. Hence, there is an urgent need for user-centric approaches that can address these challenges without requiring explicit control of the cloud environment.

In this paper, we present U-CHAMPION, a user-centric heterogeneity-aware middleware approach that automates Hadoop job provisioning and configuration in a public cloud to improve job performance and reduce the cost of leasing VM instances. However, there are several challenges in achieving heterogeneity-aware job provisioning in a public cloud.

It is challenging to develop accurate performance models for diverse Hadoop jobs running on a heterogenous cloud environment. Recent studies focused on intensive profiling of routinely executed jobs in the Hadoop environment in order to estimate their performance for various input data sizes [28]. However, such an approach is not feasible for ad-hoc jobs submitted to the system, which have unpredictable execution characteristics. To address this challenge, U-CHAMPION performs two-phase job profiling and performance modeling. In the offline phase, it applies support vector machine (SVM) regression modeling to estimate the completion time of various Hadoop jobs for different input data sizes, resource allocations, CPU models, and configuration parameters. In the online phase, it performs a lightweight profiling of ad-hoc jobs submitted to the system by using only a subset of possible configurations. Then, it applies the UV Decomposition technique to quickly estimate the job performance for all possible configurations.

U-CHAMPION's job performance models provide the foundation for making heterogeneity-aware resource allocation and configuration decisions for Hadoop jobs. However, it is significantly challenging to provision Hadoop jobs with the desired resource configurations in a public cloud. This is due to the fact that cloud providers do not allow the end-users to decide where their VM instances should be hosted. In addition, there are important cost/performance trade-offs inherent in cloud systems, which rent VM instances to users by the hour. U-CHAMPION addresses these challenges through a novel selective-instance-reacquisition technique. The main idea is to acquire new VM instances from the cloud whenever there is an expectation that doing so will result in more cost savings.

We have implemented U-CHAMPION on Amazon EC2 and evaluated its impact on Hadoop job performance and cost efficiency by using the PUMA benchmarks [3]. For comparison, we implemented AROMA [16], an automated MapReduce job provisioning system proposed recently. Experimental results demonstrate U-CHAMPION's improved accuracy in predicting ad-hoc Hadoop job performance. This is mainly due to its hardware heterogeneity awareness, and the effectiveness of the UV Decomposition approach. Furthermore, U-CHAMPION improves MapReduce job performance and reduces the cost of leasing VM instances by as much as 21%.

2 The Case Study and Motivations

Modern public clouds, such as Amazon EC2, routinely run large numbers of applications simultaneously on huge datacenters. In settings such as parallel data processing jobs, the MapReduce framework has become invaluable, allowing a relatively easy setup. However, the changing conditions within large datacenters have led to significant difficulties that are most visible in the public cloud.

2.1 Heterogeneity Characterization

Inside of a commercial datacenter, hardware is in constant flux. Servers are upgraded in sections, since fully upgrading a datacenter at once is prohibitively expensive. This has given rise to the current state, where several generations of hardware inhabit a single datacenter.

Table 1: Hardware heterogeneity in Amazon EC2.

CPU Type (Small VMs)	# of VMs	Percent of Total
US West-2 Datacenter		
E5-2650	101	82.79
E5645	21	17.21
US East Datacenter		
E5-2650	10	7.46
E5430	20	14.93
E5645	32	23.88
E5507	72	53.73

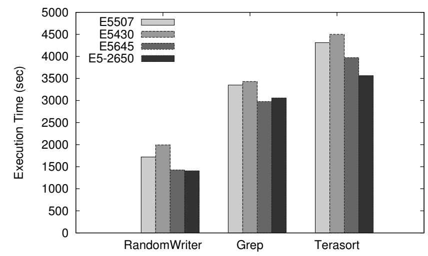

Figure 1: Impact of hardware heterogeneity on Hadoop job execution time.

We conducted a study on Amazon EC2 cloud datacenters in the US West and US East regions to measure the extent of their hardware heterogeneity. The US East datacenter in Virginia was created in 2006, while the US West-2 datacenter in Oregon started serving customers in 2011. We focused on analyzing CPU heterogeneity due to the evidence for it being the most significant source of performance variability [10, 21], and due to the ease and speed of determination (via the `cpuid` command).

The US East region shows a greater degree of hardware heterogeneity. This can be explained through the general observation that datacenters tend to grow more heterogeneous as they grow older, with newer servers being brought in to replace the older systems. Table 1 summarizes the results of our survey of the US West-2 and US East datacenters. The data was obtained by checking the CPU type of several hundred `m1.small` instances created on these datacenters on EC2. We report the number and the percentage of the VM instances running on various CPU types.

2.2 Impact of Hardware Heterogeneity

Next, we analyze the impact of Amazon EC2 hardware heterogeneity on the performance of Hadoop jobs. In this experiment, we ran three Hadoop benchmark programs (`RandomWriter`, `Grep` and `Terasort`) on Hadoop clusters of various CPU types. Each cluster consists of two VMs with the same CPU type. We ran each trial at least five times and reported the average completion times.

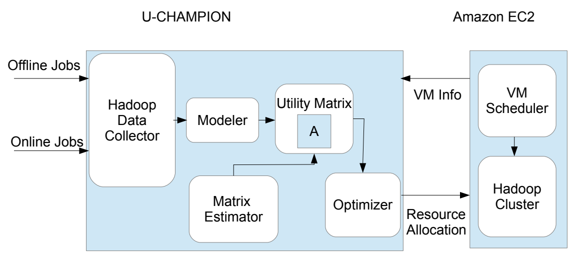

Figure 2: The system architecture of U-CHAMPION.

Figure 1 shows the job execution times of several benchmark applications running on clusters of two `m1.small` VMs. We observe that the average job execution times can vary by as much as 30% between a cluster running E5430 CPUs and one running E5-2650 CPUs. However, cloud users can not determine what CPU types will be associated with their VM instances. This motivates us to propose a user-centric heterogeneity-aware MapReduce job provisioning in the public cloud.

3 U-CHAMPION Design

3.1 Architecture

We aim to create an automated job provisioning system which integrates cluster setup optimization (cost awareness and instance selection) with heterogeneity-aware provisioning (parameter configuration, resource allocation) to improve Hadoop job performance and reduce the cost of leasing Cloud resources. The U-CHAMPION Architecture is shown in Figure 2. End users submit jobs to U-CHAMPION through a command-line interface, and our system provides appropriate configuration parameters, VM types, and underlying hardware types in order to minimize cost. With this information, a cluster is started on Amazon EC2 and the job is submitted to the master node of the cluster along with the results of the parameter optimization.

U-CHAMPION consists of three major components; the modeler, the estimator, and the optimizer.

3.2 Job Performance Modeling

In order to build accurate performance models of Hadoop jobs, we run various Hadoop benchmarks on several clusters on Amazon EC2 off line. We mine the Hadoop logs of executed jobs for execution time, input size, and various Hadoop configuration parameters. We use the *cpuid* package to obtain the node CPU configuration. This enables the creation of a database from which we can create our own performance models.

3.2.1 Support Vector Machine Models

Our system applies a powerful supervised machine learning technique to learn the performance model for each job. It constructs a support vector machine (SVM) regression model to estimate the completion time of jobs for different input data sizes, resource allocations, CPU models, and configuration parameters. SVM methodology is known to be robust for estimating real-valued functions (regression problem) from noisy and sparse training data having many attributes [7, 26]. This property of SVM makes it a suitable technique for performance modeling of complex Hadoop jobs in the Cloud environment.

We conduct stepwise regression on the data sets collected from our test-bed of virtualized Amazon EC2 Instances. For data collection, we measured the execution times of various Hadoop jobs with different input data sizes in the range of 1 GB to 50 GB, using various Hadoop parameter configurations and running on different cluster sizes of Hadoop nodes comprising of `m1.small` instances on Amazon EC2. Due to the inherent cost of running instances on EC2, we limit ourselves to `m1.small` instances in order to stretch our resources further.

U-CHAMPION incorporates hardware heterogeneity by mining data clusters for CPU type during the regression modeling. As shown in our case study, differences in CPU cause large differences in performance between seemingly identical `m1.small` instances in Amazon EC2. U-CHAMPION accounts for these differences, thereby directly increasing estimation accuracy.

3.3 Online Matrix Estimation

For ad-hoc jobs, U-CHAMPION performs a lightweight online profiling on a small portion of the input dataset with various Hadoop configuration parameters. This profiling is performed on two different CPU configurations in parallel to provide a seed of heterogeneity information. U-CHAMPION relies on online matrix estimation to obtain the complete performance model with heterogeneity information.

We apply UV Decomposition [23], a collaborative filtering technique used for matrix estimation for extremely sparse data, and which was used in the Netflix Challenge [2]. We apply it here to a similar problem, where we need to estimate the response of a job to new configuration and cluster conditions in terms of execution time by estimating based on previously collected data. UV Reconstruction has been shown to be effective for matrices where less than 5% of values are known [23].

3.4 Cost Optimization

U-CHAMPION improves the job execution time by searching for the optimal configuration and underlying hardware. It tries to provision the Hadoop cluster on the CPU type that leads to the best performance. The user has no control of VM placement in the public cloud. Thus, the optimization in U-CHAMPION has to consider both the cost of job execution and the cost of acquiring the desired CPU type.

3.4.1 Cost Estimation

The costs associated with Amazon EC2 instances are well-known and published on their website. Users are charged by the hour, and are charged for a full-hour for each partial hour used. Therefore, the cost of running a job in EC2 can be estimated as

$$c_{job} = t_{job} \cdot n_{inst} \cdot c_{inst}$$

where t_{job} is the number of hours a job takes to complete with the current cluster (rounded up to the nearest integer due to the discrete charging intervals on EC2), n_{inst} represents the number of instances, and c_{job} is the total cost of one job execution. Note that neither n_{inst} nor c_{inst} changes as a result of CPU type, so that the estimated execution time of the job is the only variable which can be optimized for a cluster of a specific size and instance type (i.e. small, medium or large standard instances).

The cost of acquiring a VM that is provisioned on a specific CPU type can be estimated by

$$e_{cpu} = \frac{c_{inst}}{p_{cpu}} \tag{1}$$

where e_{cpu} is the expected cost of obtaining a VM with given CPU type, c_{inst} is the cost of the VM instance per hour, and p_{cpu} is the probability of obtaining an instance with that CPU type from Amazon (Here we use the data provided in Table 1). We represent e_{cpu} as the expected number of VMs needed to find a certain CPU type (which is simply $1/p_{cpu}$) multiplied by the cost/hour, since Amazon charges one hour of cost upon requesting a VM.

3.4.2 Cost Saving by VM Reacquisition

Here we provide our logic for an algorithm which provides cluster optimization through selective instance reacquisition. We acquire new instances wherever we have an expectation that doing so will result in more cost savings through predicted execution time improvement than cost overhead involved in requesting additional instances and closing under-performing ones.

This leads to the examination of our tradeoff for each VM, which is

$$t_{job} \cdot c_{inst} \geq \frac{t_{job} \cdot c_{inst}}{\alpha} + e_{cpu}$$

where α is the speedup from changing CPU type. Here we state that if the total estimated cost of a VM is greater than the estimated cost of the VM with a new CPU plus the estimated cost of obtaining that CPU, then it is advantageous for us to look for higher-performing instances. The speedup α is obtained through previous results for jobs run on the various CPU types. By performing this examination on all VM instances, we are able to optimize the cost saving for the Hadoop cluster.

4 Implementation and Evaluation

4.1 Testbed

We build our testbed using Amazon EC2 service. We use the US East datacenter due to the large amount of observed hardware heterogeneity. The datacenter has four different CPU types: Intel Xeon E5-2650, Intel Xeon E5645, Intel Xeon E5507, and Intel Xeon E5430. We provisioned multiple `m1.small` virtual machine instances. Each of them have one vCPU and 1.7 GB memory. The VMs are created using the standard Amazon Machine Image (AMI) provided by *alestic.com* and installed with Ubuntu Linux 10.04.

We build Hadoop clusters using Hadoop version 1.1.2, and provision with sizes ranging from 2 to 10 slave nodes for the experiments. Each slave node is configured with one map slot and one reduce slot.

We use the PUMA benchmark suite [3] to test the performance of U-CHAMPION with representative MapReduce jobs. The PUMA benchmark contains various MapReduce benchmarks and real-world test inputs. In the experiments, we performed offline profiling on `Grep`, `Wordcount`, `Inverted Index`, and `RandomWriter` benchmarks, then performed online profiling and model estimation for the `Terasort` benchmark.

For comparison, we implemented AROMA [16]. AROMA is an automated configuration system for Hadoop parameters using machine learning to profile jobs and clustering of profiles to optimize job execution time and cost. It is hardware heterogeneity agnostic.

The output of our job models is the job execution time for a set of inputs. We used the LIBSVM library [7] to explore appropriate kernel functions and implement the SVM regression technique.

4.2 Execution Time Estimation Accuracy

First, we study the accuracy of job execution time estimation. We create a Hadoop cluster with two slave nodes on Amazon EC2. We use the `Terasort` benchmark with 20 GB input data that is generated by `RandomWriter`. We create job performance models for U-CHAMPION

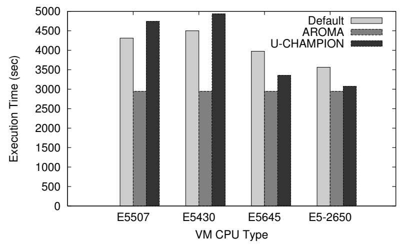

Figure 3: Estimated vs actual execution times.

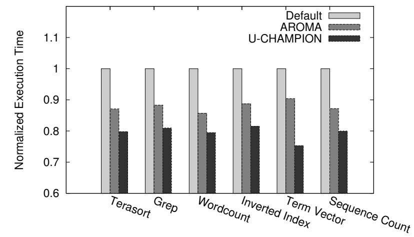

Figure 4: PUMA benchmark job execution time.

and AROMA and use them to estimate the job execution time of `Terasort` on various CPU types. We compare the estimated execution time by U-CHAMPION and AROMA to the actual execution time of the job.

Figure 3 shows the estimated job execution time of the `Terasort` benchmark on different hardware. The estimated job execution time by U-CHAMPION is 9.9%, 9.7%, 15.7%, and 13.7% different from the actual value on the E5507, E5430, E5645, and E5-2650, respectively. U-CHAMPION is able to accurately estimate execution time for different CPU types. AROMA is profiled using E5-2650 CPUs, and it provides the same estimated job execution time for different CPU types. As a result, the worst-case estimation error for U-CHAMPION is less than 16%, whereas AROMA's worst case estimation error is 35%.

4.3 Improving Job Execution Time

In this section, we evaluate U-CHAMPION's ability to reduce the overall job execution time. We build a Hadoop cluster with two slave nodes and run several PUMA benchmarks with 20 GB of input data. We use the job execution time of Hadoop with a default configuration as the baseline and compare the normalized job execution time of U-CHAMPION and AROMA. The default configuration uses heterogeneity-blind resource provisioning, meaning that we simply use whichever instances are assigned to us by Amazon.

Figure 4 shows the normalized job execution time of all benchmarks using these three approaches. The results show that U-CHAMPION outperformed the default configuration, with up to 21% shorter job execution times. U-CHAMPION provides the optimized configuration and cluster for each job, leading to this significant improvement in job execution time. In the public clouds, users are charged for VM instances by the hour. Thus, a reduction in job execution time directly results in cost savings. U-CHAMPION also outperformed AROMA, achieving up to 20% job execution time savings. U-CHAMPION achieves better performance than AROMA due to its ability to exploit the heterogeneity in the underlying hardware. U-CHAMPION not only provides an optimized job configuration, but also provisions better VM instances through selective instance reacquisition.

Table 2: Cost Optimization for `RandomWriter` Benchmark Workload.

Before Optimization	After Optimization
E5-2650 × 1	E5-2650 × 3
E5430 × 2	
E5645 × 2	E5645 × 7
E5507 × 5	
Execution time improvement	14.5%
Execution cost improvement	14.5%
Cost overhead	$1.68

4.4 Cluster Optimization vs Cost

U-CHAMPION estimates the cost of finding better-performing VM instances for a cluster, and compares it with cost-savings achieved via lower execution time due to said instances. The cost of `m1.small` VM instance is $C_{inst} = \$0.06/h$ and approximately 7.46% of our instances will be E5-2650 instances to start (see Table 1). The approximate cost of finding the best-performing CPU type (E5-2650) for `RandomWriter` is $\frac{0.06}{0.0746} = 0.804$ dollars (Eq. 1). As the execution time of our `RandomWriter` task approaches infinity, we can obtain an average of 13% and a maximum of 30% (ie, we started with only the worst-performing VM instances) cost savings by using only the best-performing VM instances.

Table 2 shows example results of our cost algorithm for a Hadoop cluster of 10 slave nodes running the *RandomWriter* with 40 GB input data. The underlying CPUs of the node before and after the algorithm is run are shown. U-CHAMPION creates 28 new VM instances to obtain the VMs with desired CPU type, therefore the cost overhead of this cluster performance enhancement is \$1.68 ($28 * C_{inst}$). Keep in mind that this provides execution time and cost improvement as long as the cluster is running. The cost will be amortized across all jobs run on this cluster until it is shut down by the user. For this example, we assume that the instances opened follow

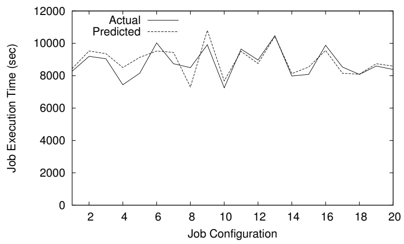

Figure 5: Prediction accuracy for an ad-hoc job for different Hadoop configurations.

the distribution which we observed in Table 1 in order to show expected results.

4.5 Adaptiveness to Ad-Hoc Jobs

Previously we showed that ad-hoc jobs submitted to the U-CHAMPION system were predicted with reasonable accuracy even in the absence of similar jobs. Here we evaluate the performance under more reasonable conditions. And, we show that the model generated for the `Terasort` benchmark by UV Decomposition remains useful under a variety of job configurations.

This experiment assumes that a `Terasort` workload with 20GB of input data (generated by TeraGen) has been submitted to the system. We also assume an 8 VM cluster, and that several other benchmarks (`Wordcount`, `Grep`, `Inverted Index`, etc) have been profiled offline. In this case, we use the methodology described in section 3.3 to create a model for `Terasort`. Figure 5 shows the prediction accuracy for 20 different Hadoop configurations using the new model. We see prediction error here of less than 17%.

5 Related Work

Recent studies have focused on improving the performance of applications in clouds [5, 6, 8, 12, 14, 24] through elastic resource allocation and VM scheduling. Paragon [8] implements a heterogeneity-aware job scheduling system using a Singular Value Decomposition (SVD) technique similar to U-CHAMPION, but considers scheduling only single-node applications and requires full control of the cloud environment, making it very hard to use for a user in the public cloud.

Users of the public cloud only have limited information about the cloud environment and have no control of the hardware their VMs run on. U-CHAMPION, along with other user-centric research [16, 19], uses the limited information that is available in order to improve the decisions made by *one* user.

Hardware heterogeneity is a prevalent issue in public clouds [10, 17, 21]. Recent studies show that it is feasible to leverage the hardware heterogeneity to improve the performance of applications [9–11, 20, 21].

There are also some works focusing on improving Hadoop performance by reducing the delay due to shuffle and straggler tasks [13, 18, 30]. Park *et al.* proposed a novel VM reconfiguration approach that is aware of the data locality of Hadoop [22]. Guo *et al.* proposed and implemented iShuffle [13], a user-transparent shuffle service that pro-actively pushes map output data to nodes via a novel shuffle-onwrite operation and flexibly schedules reduce tasks considering workload balance.

There is a rich set of research focused on the parameters and performance of Hadoop clusters. Jiang *et al.* [15], conducted a comprehensive performance study of Hadoop and summarized the factors that can significantly improve Hadoop performance. Verma *et al.* [28, 29], proposed a cluster resource allocation approach for Hadoop. AROMA [16] provides a novel framework for automated parameter estimation and cluster resource provisioning in order to maximize job performance in a given cluster.

6 Conclusion

U-CHAMPION is proposed and developed to enable a user-centric and heterogeneity-aware MapReduce job provisioning in the public cloud. It addresses the unique challenges imposed by the public cloud environment through a novel selective-instance-reacquisition technique. This technique applies our proposed optimization algorithm to acquire new VM instances if it results in more cost savings. Furthermore, U-CHAMPION is able to make accurate performance prediction of ad-hoc Hadoop jobs through its UV Decomposition technique and by the incorporation of hardware heterogeneity-awareness in job performance modeling. Extensive evaluation of U-CHAMPION on Amazon EC2 Cloud with representative benchmark applications demonstrated its improved performance prediction accuracy as compared to a heterogeneity-unaware approach. Furthermore, the results showed its ability to improve Hadoop job performance and reduce the cost of leasing the Cloud resources by up to 21%.

Our future work will extend U-CHAMPION to a multi-user environment for mitigating performance interference.

Acknowledgement

This research was supported in part by U.S. NSF CAREER award CNS-0844983, research grants CNS-1320122 and CNS-1217979.

References

[1] Apache Hadoop Project. `http://hadoop.apache.org`.

[2] Netflix Prize. `http://www.netflixprize.com`.

[3] PUMA: Purdue mapreduce benchmark suite. `http://web.ics.purdue.edu/~fahmad/benchmarks.htm`.

[4] AHMAD, F., CHAKRADHAR, S. T., RAGHUNATHAN, A., AND VIJAYKUMAR, T. N. Tarazu: Optimizing mapreduce on heterogeneous clusters. In *Proc. of the ACM Int'l Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)* (2012).

[5] BEN-YEHUDA, O. A., BEN-YEHUDA, M., SCHUSTER, A., AND TSAFRIR, D. The resource-as-a-service (raas) cloud. In *Proc. of the USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)* (2012).

[6] BU, X., RAO, J., AND XU, C. Interference and locality-aware task scheduling for mapreduce applications in virtual clusters. In *Proc. of the ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC)* (2013).

[7] CHANG, C.-C., AND LIN, C.-J. LIBSVM: A library for support vector machines. *ACM Trans. Intelligent System Technology* (2011).

[8] DELIMITROU, C., AND KOZYRAKIS, C. Paragon: Qos-aware scheduling for heterogeneous datacenters. In *Proc. of the ACM Int'l Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)* (2013).

[9] DELIMITROU, C., AND KOZYRAKIS, C. Quasar: Resource-efficient and qos-aware cluster management. In *Proc. of the ACM Int'l Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)* (2014).

[10] FARLEY, B., JUELS, A., VARADARAJAN, V., RISTENPART, T., BOWERS, K. D., AND SWIFT, M. M. More for your money: exploiting performance heterogeneity in public clouds. In *Proc. of the ACM Symposium on Cloud Computing (SoCC)* (2012).

[11] GUEVARA, M., LUBIN, B., AND LEE, B. C. Market mechanisms for managing datacenters with heterogeneous microarchitectures. *ACM Trans. on Computer Systems 32*, 1 (Feb. 2014), 3:1–3:31.

[12] GUO, Y., LAMA, P., RAO, J., AND ZHOU, X. V-cache: Towards flexible resource provisioning for clustered applications in iaas clouds. In *Proc. IEEE Int'l Parallel and Distributed Processing Symposium (IPDPS)* (2013).

[13] GUO, Y., RAO, J., AND ZHOU, X. iShuffle: Improving hadoop performance with shuffle-on-write. In *Proc. of the USENIX Int'l Conference on Autonomic Computing (ICAC)* (2013).

[14] JALAPARTI, V., BALLANI, H., COSTA, P., KARAGIANNIS, T., AND ROWSTRON, A. Bridging the tenant-provider gap in cloud services. In *Proc. of the ACM Symposium on Cloud Computing (SoCC)* (2012).

[15] JIANG, D., OOI, B. C., SHI, L., AND WU, S. The performance of MapReduce: an in-depth study. *Proc. VLDB Endowment* (2010).

[16] LAMA, P., AND ZHOU, X. AROMA: automated resource allocation and configuration of mapreduce environment in the cloud. In *Proc. of the ACM Int'l Conference on Autonomic computing (ICAC)* (2012).

[17] LEE, G., CHUN, B., AND RANDY, H. Heterogeneity-aware resource allocation and scheduling in the cloud. In *Proc. of the USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)* (2011).

[18] LI, X., WANG, Y., JIAO, Y., XU, C., AND YU, W. CooMR: Cross-task coordination for efficient data management in mapreduce programs. In *Proc. of the Int'l Conference for High Performance Computing, Networking, Storage and Analysis (SC)* (2013).

[19] LIM, S.-H., HUH, J.-S., KIM, Y., SHIPMAN, G. M., AND DAS, C. R. D-factor: a quantitative model of application slow-down in multi-resource shared systems. In *Proc. ACM SIGMETRICS* (2012).

[20] MARS, J., AND TANG, L. Whare-map: Heterogeneity in "homogeneous" warehouse-scale computers. In *Proc. of the ACM Int'l Symposium on Computer Architecture (ISCA)* (2013).

[21] OU, Z., ZHUANG, H., NURMINEN, J. K., YLÄ-JÄÄSKI, A., AND HUI, P. Exploiting hardware heterogeneity within the same instance type of amazon ec2. In *Proc. of the USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)* (2012).

[22] PARK, J., LEE, D., KIM, B., HUH, J., AND MAENG, S. Locality-aware dynmaic vm reconfiguration on mapreduce clouds. In *Proc. of the ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC)* (2012).

[23] RAJARAMAN, A., AND ULLMAN, J. *Textbook on Mining of Massive Datasets*. 2011.

[24] RAO, J., WANG, K., ZHOU, X., AND XU, C. Optimizing virtual machine performance in NUMA multicore systems. In *Proc. of the IEEE Int'l Symposium on High Performance Computer Architecture (HPCA)* (2013).

[25] REISS, C., TUMANOV, A., GANGER, G. R., KATZ, R. H., AND KOZUCH, M. A. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In *Proc. of the ACM Symposium on Cloud Computing (SoCC)* (2012).

[26] SCHÖLKOPF, B., SMOLA, A. J., WILLIAMSON, R. C., AND BARTLETT, P. L. New support vector algorithms. *Neural Comput.* (2000).

[27] SCHWARZKOPF, M., MURRAY, D. G., AND HAND, S. The seven deadly sins of cloud computing research. In *Proc. of the USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)* (2012).

[28] VERMA, A., CHERKASOVA, L., AND CAMPBELL, R. H. ARIA: automatic resource inference and allocation for mapreduce environments. In *Proc. of the ACM Int'l Conference on Autonomic Computing (ICAC)* (2011).

[29] VERMA, A., CHERKASOVA, L., AND CAMPBELL, R. H. Resource provisioning framework for mapreduce jobs with performance goals. In *Proc. of the ACM/IFIP/USENIX Int'l Conference on Middleware* (2011).

[30] WANG, Y., QUE, X., YU, W., GOLDENBERG, D., AND SEHGAL, D. Hadoop acceleration through network levitated merge. In *Proc. of the Int'l Conference for High Performance Computing, Networking, Storage and Analysis (SC)* (2011).

[31] WOLF, J., RAJAN, D., HILDRUM, K., KHANDEKAR, R., KUMAR, V., PAREKH, S., WU, K., AND BALMIN, A. Flex: a slot allocation scheduling optimizer for mapreduce workloads. In *Proc. of the ACM/IFIP/USENIX Int'l Conference on Middleware* (2010).

[32] ZAHARIA, M., KONWINSKI, A., JOSEPH, A. D., KATZ, R., AND STOICA, I. Improving mapreduce performance in heterogeneous environments. In *Proc. of the USENIX Conference on Operating Systems Design and Implementation (OSDI)* (2008).