# Research on Scale-Out Workloads and Optimal Design of Multicore Processors

**Qiong Wang, Li Shen and Zhiying Wang**

**Abstract** In recent years, cloud computing has been emerging as an infrastructure of online services. Most of the applications deployed on the data center have the typical scale-out features, such as Google search engine, MapReduce, and media streaming. However, while social demand for cloud computing continues to grow, the infrastructure in the data center cannot meet the needs. The inherent characteristics of scale-out workloads place them into a distinct workload from desktop, parallel, and traditional server workloads. Therefore, data center efficiency should be improved by matching the processor design to the needs of the scale-out workloads. In this paper, we test some representative benchmarks of scale-out workloads and find out their performance under different core counts and CPU frequencies. Our work verifies its needs both in low latency and in high throughput. Moreover, we analyze the results and propose several ways to improve the performance of multicore processors.

**Keywords** Scale-out workloads · Throughput · Latency · Multicore processors

## 1 Introduction

Driven by application, semiconductor technology, and architecture improvements, multicore processors have been widely used in various fields from high-performance computing to servers, desktop computing, and embedded systems.

Q. Wang (✉) · L. Shen · Z. Wang
State Key Laboratory of High Performance Computing, School of Computer, National University of Defense Technology, 410073 Changsha, China
e-mail: wangqiong@nudt.edu.cn

L. Shen
e-mail: lishen@nudt.edu.cn

S. Patnaik and X. Li (eds.), *Proceedings of International Conference on Soft Computing Techniques and Engineering Application*, Advances in Intelligent Systems and Computing 250, DOI: 10.1007/978-81-322-1695-7_19, 

Meanwhile, in some fields which have higher standard for throughput, power, and performance, multicore processors have been playing a more important role. For instance, several multicore accelerators have been used in high-performance computing such as NVidia GPU and Intel Xeon Phi. In the latest TOP 500 ranking list [1], most of the supercomputers ranking the top 10 have adopted the "multi-core CPU + many core accelerator" heterogeneous systems such as Tianhe 2, Titan, and Stampede. Moreover, the servers based on Tilera Gx [2] series multi-core processors can fill the needs of digital media, Internet communication, and other same fields. Undoubtedly, multicore processors will be the first choice of high-performance computing in the long run. However, the diversification in the type of applications brings new challenge to the design of multicore processors. One of them is scale-out workloads. Most of such applications deployed on the data center, such as data serving, MapReduce, media streaming, SAT solver, Web front end, and Web search, all have the typical scale-out features. Along with the data center becoming the economic infrastructure as well as transportation and energy, those applications are playing a vital role in our life.

According to the traditional design methods, modern processors can be classified into two types. One is processor which has less number of cores but complex in order to decrease the latency such as Intel/AMD multicore processors. Another type has more cores but simpler to improve the throughput such as Sun Niagara processors. However, both general-purpose and traditional server processors are all targeting for the characteristics of scale-up workloads. These workloads pursue either a high performance of single threads or a high throughput. Moreover, most of the multicore processor designs follow a trajectory that benefits scale-up workloads, which means, to meet the increasing high-performance demand by adding more computational resources in a single node. However, the scale-out workloads deployed in the data center have distinct characteristics, which bring challenges to the design and optimization of multicore processors. Distinct from existing desktop, parallel, and traditional server workloads, scale-out workloads have some brand new features such as high I-cache miss ratio, low instruction level parallel (ILP), large work sets, and low demand for on-chip and off-chip bandwidth.

In this work, our tests show that scale-out workloads have both needs in single thread efficiency and the number of threads, which on the other side verifies that exiting modern processors cannot support the scale-out workloads efficiently. We test and analyze several representative scale-out workloads' benchmark and find out the influences of core numbers and frequencies on the latency and throughput, proving that scale-out workloads have both needs in single thread efficiency and number of thread numbers. One step further, we propose some methods to optimize the existing multicore processors based on the results to match the needs of scale-out workloads.

The rest of the paper is organized as follows. Section 2 introduces the representative scale-out workloads we test to. Section 3 performs the methodology and test results. Section 4 proposes several methods to the optimal design of multicore processors for scale-out workloads. Section 5 summarizes the related works, and Sect. 6 concludes the work carried out in the paper.

## 2 Scale-out Workloads

As cloud computing becomes ubiquitous, the number of scale-out workloads based on cloud platform increases at the same time. They all share some similar characteristics as follows: (1) based on the large amount of data sets in the clusters; (2) handle independent user requests having no interact data; and (3) designed specifically for use in cloud infrastructure.

CloudSuite [3] is a benchmark suite of scale-out workloads. It is chosen based on the popularity of online services. It consists of eight most popular application benchmarks in the data center, including data serving, data analytics, Web serving, Web search, media streaming, data caching, graphic analytics, and software test. Those benchmarks operate on real software stack and represent the real system configuration. Our work has chosen two of them to introduce and test.

*Data Serving* Most of the online services use NoSQL as its huge data storage such as Cassandra, HBase, and PNUTS. They split data into fragments and scale out to clusters. Cassandra is a mixed no-relation database. One of its features is that it is not a database in fact, but a distributed network service that consists of a lot of data nods. For a write operation to the database, the data will be replicated to other nodes. And also a read operation will be routed to some node in the cluster. For a Cassandra cluster, it is easy to improve the performance through adding more nodes. Yahoo! Cloud Serving Benchmark [4] (YCSB) is a framework to test data storage system, which provides test interface for various popular data service system. Its main purpose is to test the cloud service infrastructure to promote comparing emerging cloud data service system. Data serving is a benchmark that consists of YCSB and Cassandra. YCSB sends read or write operation to Cassandra and tests its performance.

*Data Analytics* Mahout is a framework of machine learning and data mining. Building on Hadoop distinguishes itself from other open source data mining software. It provides some extensible classical algorithm realization in data learning field to support researchers to create intelligent applications. Hadoop is an open source software framework that supports data-intensive distributed applications. It supports the running of applications on large clusters of commodity hardware. It has the reliability, scalability, efficiency, and high fault tolerance features. Since it assumes that the computation elements and storage can fail, it maintains several working data copy to ensure the reconstruction of failure nodes. Moreover, it does its work in parallel to speed up the process. Its bottom layer consists of Google File System (GFS) and Google's MapReduce. Data analytics is a Mahout implementation of machine learning and data mining constructed on Hadoop cluster.

## 3 Experiment and Analysis

Scale-out workloads deployed on the cloud platform process hundreds and thousands of independent requests from user terminals, which have no sharing data. Therefore, the data center should create enough threads to response and handle the messages. So we conjecture that the more the cores and parallel threads, the higher the throughput we can get. We can verify it by testing the relationship between the number of cores and the application performance. Meanwhile, as users, they would not want to wait too long to get response. Therefore, the performance of a single thread cannot be ignored.

According to the analysis above, we test the needs of two scale-out workloads benchmarks in the throughput and latency respectively. We conduct our study in an Intel Xeon E5350 machine with eight cores and an Intel SandyBridge with four cores. Both of them use CentOs 6.4 with 2.6.2 kernel.

### *3.1 Throughput*

The definition of the throughput is the number of user requests completed within a unit time. Under the same number of user requests, the less time it takes, the higher the throughput. Therefore, we can get the relationship between throughput and cores through testing the time under different number of cores. In the experiment of testing the needs of scale-out workloads in throughput, we increase the number of cores gradually and obtain the time of dealing with all the requests. Figure 1 shows the execution time of data serving and data analytics, respectively, under different number of cores. From the chart, we can see that the time decreases with core count. To make it clearer, we calculate the speedup normalized to one core as shown in Fig. 2. In Fig. 2a, when core number is above eight, the curve has the trend to continue increasing. But for data analytics, the speedup tends to be flat when it reaches the point of eight cores. The reason is that in addition to a lot of threads created to map and reduce, it has hundreds of terabytes of data to deal with.

As for these two applications, under same user requests, the speedup increases as the core number increases, which shows that scale-out workloads have a high demand for core numbers. One of the reasons is that it is determined by the huge amount of users it target to when it is designed, and the other is by its inherent characteristics. As to the scale-out workloads deployed on the cloud platform, it has to create large numbers of threads to response to the requests sent from the users around the world. By increasing the core numbers, the number of hardware threads increases and the throughput increases as well, which reflects its needs in core amount. Modern existing processors are designed specifically for scale-up workloads, which cannot support scale-out workloads efficiently. First of all, almost all the processors develop ILP through out-of-order (OoO) implementation. Researchers increase the depth of assembly line and enlarge the instruction

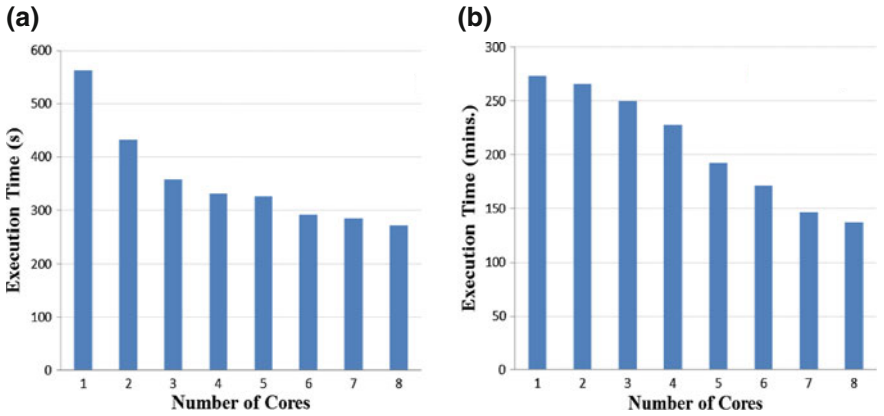


**Fig. 1** Execution time varying the number of cores. **a** Data serving. **b** Data analytics

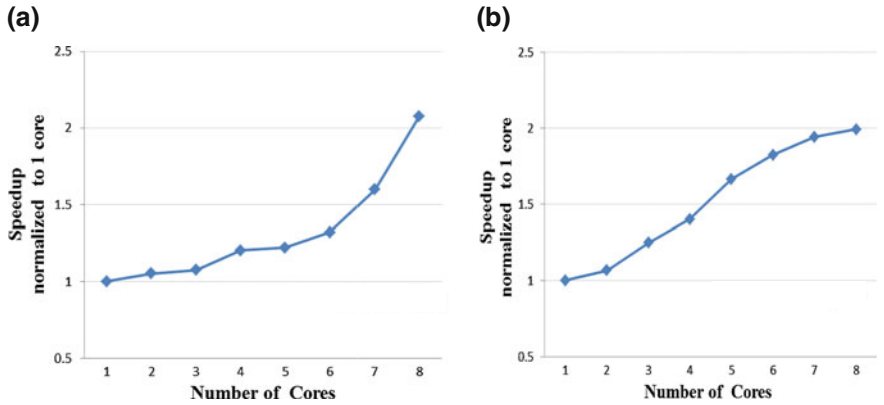


**Fig. 2** Speedup varying the number of cores. **a** Data serving. **b** Data analytics

window to develop more ILP, but get trivial benefits since scale-out workloads have limited ILP. Deep assembly line and OoO implementation request large number of hardware to support such as multiple branch prediction, instruction schedulers, forwarding paths, many-port register banks, load–store queues (LSQ), ALUs, reorder buffers (ROB), and other on-chip structures. However, within the limit chip area, the more complex the core is, the less the number of cores and the less the hardware threads. Since scale-out workloads need the multicore processors which are highly computation intensive and power efficient, designers have to distribute the resources to the processors, cache on-chip, and core interconnect rationally.

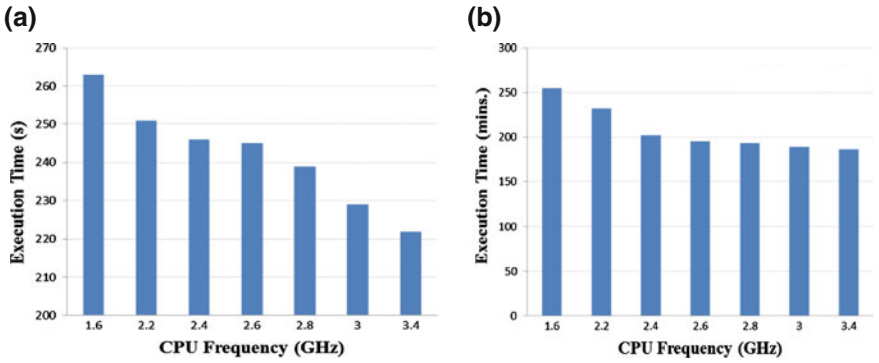


**Fig. 3** Execution time varying frequency. **a** Data serving. **b** Data analytics

## *3.2 Latency*

In the experiment of examining the scale-out workloads' needs in single thread performance, we test the time under different frequencies since CPU frequency has influence on its computation speed, and therefore, we can obtain the relationship between core computing performance and the applications. Figure 3 demonstrates the execution time of data serving and data analytics under different frequencies. Figure 4 plots data serving and data analytics speedup under different frequencies normalized to the lowest 1.6 GHz. In Fig. 4a, it shows that for data serving, the speedup increases as the frequency increases, but the highest point does not exceed 1.2 yet. For data serving benchmark, we can find that the highest point in Fig. 4a is not as much as in Fig. 1a, which to some extent shows that the number of cores has more influence on this application than core computation performance. The reason is that Cassandra needs huge amount of threads to respond but relatively simple process to handle. In Fig. 4b, even under the highest frequency 3.4 GHz, the highest point is merely 1.4, which shows limited benefit from high frequency. Above all, the speedup increases when the frequency becomes higher, but not sharp, which means that the better core performance brings trivial benefits to these applications. Therefore, existing deep assembly line and complex core occupies the die area but cannot benefit scale-out workloads. When devise a multicore processor for scale-out workloads, we cannot choose either from the complex core designed for computer intensive applications or simple core designed for traditional server applications.

Scale-out workloads have high demand both in response latency and in throughput, but these two, to some extent, contradict each other. In order to shorten the response latency, computation resources in the core should be increased (such as the number of computation units), which would take more die area at the same time. And in the interest of high throughput, more cores (or hardware threads supported by per-core) should be added in the processors, which would decrease

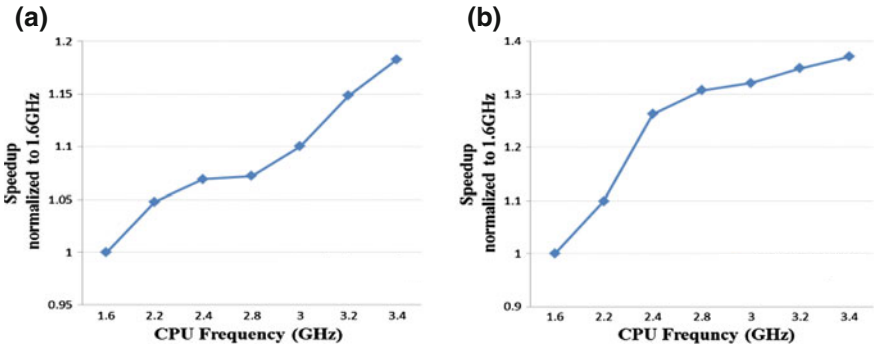


**Fig. 4** Speedup varying frequency. **a** Data serving. **b** Data analytics

the per-core areas on chip. Therefore, researchers need to design a multicore architecture, specifically for the scale-out workloads on the limited die area. The architecture researchers have summarized some design experience in pursing the best performance under given area or (and) power constraints.

By far, the automatic design process based on fine-grained accelerator has been built, which can adjust existing instruction set and design a core that meets the demand of single thread application and other corresponding tools. For instance, application-specific instruction set architecture [5] (ASIP) can automatically do the confirmation of extensive instruction, the design of the function unit, and the modification of software tools. Scale-out workloads contain a lot of data level parallel so that performance can be improved by SIMD. Fine-grained accelerator can benefit scale-out workloads in area of function unit, power, and latency and meet its needs in latency, throughput, and power.

## 4 Optimal Design of Multicore Processors for Scale-out Workloads

Scale-out workloads have huge data sets and complex instruction stream and need high throughput and low latency. However, most existing multicore processors cannot meet its needs. Under such condition, we need to design and optimize the multicore architecture, specifically for scale-out workloads. In future work, we should start from following aspects.

*Microarchitecture* Most of the scale-out workloads are online services demanding for short response latency and high throughput. However, they cannot fully take advantage of deep assembly line or OoO core due to their limited ILP [6], which conversely wasted many transistor resources on chip. Moreover, in order to improve the throughput, more cores (or threads) should be added, which put constraints on the die area. Therefore, constraints and demands should be considered at the same time to provide enough cores and improve the thread

performance. Scale-out workloads contain lots of data level parallel so that performance can be improved by SIMD. Fine-grained accelerator can benefit scale-out workloads in area of function unit, power, and latency and meet its needs in latency, throughput, and power.

*Cache hierarchy* Scale-out workloads do not have obvious temporality of instruction, leading to a high I-cache miss ratio. Moreover, the instruction working set considerably exceeds the last-level cache, which causes high access latency. We have to find out an optimal cache hierarchy for scale-out workload and consider its impact on power efficiency.

*Network-on-chip* (*NoC*). Threads created by scale-out workloads are mostly independent and have few data coherence operations, which makes it need no high bandwidth interconnect on-chip. Simple crossbar construction is enough for the communication between cores when the number of cores is small. However, with the improvement in cores and caches on-chip, the number of cores on the chip can be increased to tens or hundreds. Therefore, a well-suited on-chip interconnect for scale-out workloads should be considered. Dynamic reconfigurable topology NoC should be a good option since it supports reconfiguration of network paths, which can gain a high throughput and decrease network latency at the same time.

*Memory controller* Modern processors generally have integrated high-performance memory controller as a medium between processors and memory to overcome the timing and resource constraints brought by the storage devices and to realize the access efficiency to the memory. Current DDRx memory controllers basically adopt fixed hardware logic units including complex address mapping logic, request scheduling logic, power management, and updated algorithms. However, fixed hardware logic implementations cannot fill the needs of scale-out workloads. We propose to use a programmable memory controller, which can enhance the memory system ability of adjusting to the various applications.

## 5 Related Work

Existing multicore processors are designed specifically for scale-up workloads satisfying the emerging needs by adding more computation resources. However, scale-out workloads deployed on the data center have shown some distinct characteristics that bring new challenge to the design and optimization of multicore processors. Kgil et al. [7] show that modern processors is power inefficiently for the Web applications which emphasize more on high throughput. As scale-out workloads become ubiquitous, researchers start to analyze its features. Some of them are from system levels [4, 8–10], and the others are from the microarchitecture [11, 12] levels. There are some methods of design oriented to applications in multicore processors and use it in the design of commercial processors, but no matter the product construction itself or usage efficiency all have space to improve. More importantly, those methods are all aimed at improving the scale-up workloads, which cannot be used to design scale-out workloads' processors directly.

Until now, researchers have found some preliminary characteristics of scale-out workload and proposed some custom-made strategy of multicore processors gaining some certain benefits, but there is still a long way to summarize the relative design and optimization theory. Therefore, we need to analyze the scale-out workloads and explore the design and optimization method of multicore processors. Ferdman et al. [6] test and analyze the I-cache miss ratio, ILP, and bandwidth usage of scale-out workloads. They point out that the huge amount of data set exceeds the size of cache on-chip, but existing hierarchical cache takes a lot of die area but cannot support its efficient implementation. Oh et al. [13] show that the time spent on the last level occupies half of the data stall, which means that existing cache hierarchy is not rational. Our results corroborate these findings, showing that we should increase more cores to improve the throughout on the limited die area.

## 6 Conclusions

Scale-out workloads have both demands in low latency and in high throughput, and its inherent characteristics distinct from traditional workloads bring new challenges and opportunities to the design and optimization of multicore processors. As scale-out workloads become ubiquitous, its impact on social life is growing as well. However, existing multicore processors are designed specifically for scale-up workloads, which cannot meet its needs in computation density and power efficiency. To design multicore processors for scale-out workloads becomes a challenge researchers confronted to. In this work, we test several representative benchmarks of scale-out workloads and prove that scale-out workloads have the needs both in low latency and in high throughput, which provides experience in the multicore architecture design for scale-out workloads in the further study. And we propose several aspects of optimization and design of multicore architecture, specifically for scale-out workloads.

**Acknowledgments** This work is supported by the National Basic Research Program of China (863 Program) under Grant No. 2012AA0-10905, National Natural Science Foundation of China under Grant No. 61272143.

## References

1. TOP500 supercomputer sites. http://www.top500.org (2013)
2. Tilera Inc. http://www.tilera.com (2013)
3. Cloudsuite. http://parsa.epfl.ch/cloudsuite (2013)
4. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with YCSB. In: Proceedings of the 1st ACM Symposium on Cloud Computing, June 2010

5. Keutzer, K., Malik, S., Newton, A.R.: From ASIC to ASIP: the next design discontinuity. In: ICCD'02, pp. 84–90 (2002)
6. Ferdman, M., Adileh, A., Kocberber, O., et al.: Clearing the clouds, a study of emerging scale-out workloads on modern hardware. In: ASPLOS (2012)
7. Kgil, T., D'Souza, S., Saidi, A., Binkert, N., Dreslinski, R., Mudge, T., Reinhardt, S., Flautner, K.: PicoServer: using 3D stacking technology to enable a compact energy efficient chip multiprocessor. In: Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems, Oct 2006
8. NVIDIA Tesla Computing Processor. http://www.nvidia.com/docs/IO/43395/NV_DS_Tesla_C1060_US_Jan10_lores_r1.pdf
9. Li, A., Yang, X., Kandula, S., Zhang, M.: CloudCmp: comparing public cloud providers. In: Proceedings of the 10th Annual Conference on Internet Measurement, Nov 2010
10. Kozyrakis, C., Kansal, A., Sankar, S., Vaid, K.: Server engineering insights for large-scale online services. IEEE Micro **30**(4), 8–19 (2010). (July–Aug)
11. Janapa Reddi, V., Lee, B.C., Chilimbi, T., Vaid, K.: Web search using mobile cores: quantifying and mitigating the price of efficiency. In: Proceedings of the 37th Annual International Symposium on Computer Architecture, June 2010
12. Tang, L., Mars, J., Vachharajani, V., Hundt, R., Soffa, M.L.: The impact of memory subsystem resource sharing on datacenter applications. In: Proceeding of the 38th Annual International Symposium on Computer Architecture, June 2011
13. Oh, T., Lee, H., Lee, K., Cho, S.: An analytical model to study optimal area breakdown between cores and caches in a chip multiprocessor. In: Proceedings of the IEEE Computer Society Annual Symposium on VLSI, May 2009