1

Scalable Computing and Communications: Past, Present, and Future

Yanhui Wu, Kashif Bilal, Samee U. Khan, Lizhe Wang, and Albert Y. Zomaya

1.1 SCALABLE COMPUTING AND COMMUNICATIONS

Scalability is a paradigm that can adapt to the need of computing requirements of the underlying applications and users. Scalability [1, 2] is also a desirable quality for a network, process, website, or business model. In terms of hardware, a scalable computer system may begin with one node, but more nodes can be added as and when there is a need for more computing capabilities. Scalability, when sold with IT equipment or software, is a feature to convince high-growth businesses that the future needs can be accommodated easily and without recourse to expensive machine replacement or staff retraining. However, a scalable system need not be at one physical address. The ease of availability of high-speed networks and powerful computers has led to the emergence of two computing trends: (1) cluster computing [3, 4] and (2) grid computing [5–9]. Geographically, remote desktop computers, storage systems, data sources, scientific instruments, and clusters can be combined into what are known as computational grids.

Cloud computing [10–17] is an emerging paradigm in which users export data and applications (or computations) to the "cloud" (a remote set of machines) and then access the data or application in a simple and pervasive way. However, the aforementioned is a classic example of central processing. Interestingly, about 50

Scalable Computing and Communications: Theory and Practice, First Edition. Edited by Samee U. Khan, Albert Y. Zomaya, and Lizhe Wang.

years ago, a similar situation arose when time-sharing computing servers catered to multiple users. The above-mentioned paradigm took a major shift with the introduction of personal computers (PCs) that allowed data and programs to be stored and processed locally. Most certainly, the evolution of the cloud computing paradigm is not exactly the same as the last cycle (time-sharing to PC) in the computing history, simply due to the fact that computation resources are not scarce anymore. We must understand that the clouds have emerged due to the need to build complex IT infrastructures that demand management of various software installations, configurations, updates, and heterogeneous resources. Computing resources and other hardware are becoming obsolete very rapidly. Therefore, outsourcing computing platforms is the most viable solution for users who would like to handle complex IT infrastructures.

Cloud computing is very rapidly evolving and there seems to be no widely accepted definition. Based on our experience, we proposed an early definition of cloud computing as follows: "A computing cloud is a set of network enabled services, providing scalable, QoS guaranteed, normally personalized, and inexpensive computing infrastructures on demand, which could be accessed in a simple and pervasive way." However, a more rigid and concise definition may be "A cloud computing paradigm is simply put a data center [18–25] that is being accessed by an application programming interface (API)."

Because cloud computing is the state of the art of computing paradigms [26–29], it is imperative to discuss issues related to scalability in the context of clouds. Looking to the future, cloud computing standardization efforts may well mirror what we observed for the standardization efforts toward the Internet.

For clouds, many standards development organizations (SDOs), consortia, and trade associations are busy creating cloud computing standards. As a consequence, there may likely be multiple standards in some areas, while other areas may miss standards entirely. It is this challenge that the IEEE Cloud Computing Initiative aims to address.

The IEEE P2301 [30] is a metastandard, a set of profiles consisting of other standards, publications, and guidelines from many organizations. The metastandard will provide profiles of existing and in-progress cloud computing standards in critical areas, such as applications, portability, management, and interoperability interfaces, file formats, and operation conventions. With capabilities logically grouped so that it addresses different cloud audiences and personalities, the IEEE P2301 metastandard will provide an intuitive roadmap for cloud vendors, service providers, users, developers, and other key stakeholders. When completed, the standard will aid users in procuring, developing, building, and using standards-based cloud computing products and services, enabling better portability, increased commonality, and greater interoperability across the industry. The IEEE P2302 is like any other standards except that it is focuses on inter- and intracloud problems, and there are no other efforts that we know of that are close to such an effort. The IEEE P2302 [31] essentially defines the topology, protocols, functionalities, and the governance required for the reliability of inter- and intracloud interoperability and federation. The standard will help build an economy of scale among cloud product and service providers, which will remain transparent to the users and applications. With a dynamic infrastructure supporting evolving cloud business models, the IEEE P2302 standard is an ideal platform for fostering growth and for improving competitiveness. It will also

address fundamental, transparent interoperability and federation, much in the way as SS7/IN did for the global telephony system and as naming and routing protocols did for the Internet.

As the complexity and the size of the underlying systems increase, so does their demand for scalability. Therefore, the issues pertaining to scalability must be addressed head on. It is extremely difficult to come up with the top five problems within the general area of scalable computing. However, based on our experience, expertise, and discussions, the following five problems (in no particular order) seem to be the most critical issues within the realm of scalable computing.

1. *Data-Intensive Scalable Computing.* It is relatively inexpensive to gather digital data compared to a few decades ago. Therefore, more and more applications are emerging that are so-called data guzzlers. Examples of the aforementioned may include medical imaging, surveillance data, inventory data of shopping megastores, and location services [32–34]. The influx of information entails efficient and effective storage, processing, and transportation of such data. This is an uncharted territory for scalable computing. It would be unwise to even cap the scale of such a data influx to exascale computing. The grand solution to the above-mentioned problem will include directions from domains, such as scheduling, data placement, data management, operating systems, databases, programming languages, computer communications, computer architectures, disk storage devices, and middleware.
2. *Scalable Programming Paradigm.* A critical bottleneck in accessing the underlying massive-scale computing devices is the programming interface. Therefore, efficient and effective programming modules, more specifically, parallel programming libraries, must be developed to cater to the massive scales of the new generation of large-scale computing systems [35, 36]. Certain key issues in the aforementioned domain will be (1) reducing bandwidth requirements as much as possible; one methodology could be by introducing the use of mixed precision or storing data in 32-bit arrays wherever possible; and (2) rewriting low-level kernels as stateless functions with large enough granularity to keep a single-instruction multiple-data (SIMD) core busy and small enough that there is a large volume of simultaneous function calls to execute.
3. *High-Dimensional and Highly Dynamic Problem Sets.* More than often, computer algorithms are constructed for a deterministic or a bounded input. However, as the complexity of the underlying system increases, interactions among system components may not be fully captured by a bounded input. Therefore, various methodologies must also be incorporated that can adjust the system resource manager in an autonomic fashion to react to system uncertainties [37]. A plausible solution is to model and simulate the system under various system parametric fluctuations. However, all of the advanced modeling and simulation techniques quickly increase the problem size and parallelism, often by an order of magnitude, and large problems can easily exceed the computing capacity of the currently available computing systems. The simplest of the simulation approaches are "black box" in nature, which does not require a true exascale system (instead requiring a cluster of petascale systems). However, more advanced methods (termed embedded methods)

rely on a tightly coupled aggregation of forward problems and require true exascale systems. The challenge with embedded methods is that they require the transformation of an application into a "subroutine" because embedded methods need to invoke "forward solve" as a function. Most applications are not designed with this mindset, so this transformation will be challenging.

4. *Fault-Resilient Application Environment.* If hardware fault predictions are accurate, exascale systems will have very high fault rates and will, in fact, be in a constant state of restore and recovery. "All of the nodes are up and running," which is our current sense of a well-functioning scalable system, will become infeasible. Instead, we may always have a portion of the system dead, a portion that is dying, a portion continuously producing faulty results, a portion that is coming back to life, and a final, hopefully larger, portion that is computing fast and accurate results. The current hardware and software environments are not well prepared for the above-mentioned "kind of stable" system. In fact, the only reliable, portable resilience mechanism we have is checkpoint–restart. Although there have been many research efforts to achieve fault tolerance, much of this work has been focused on a single layer in the hardware and software stack, without sufficient consideration of the whole set of requirements. One of the biggest needs we have in resilient computing research is an increased effort to include the full vertical scope of the software and hardware stacks into our design discussions. Moreover, we need a full-featured environment to probe the system, to make decisions based on system state, and to recover from system faults, both hardware and software. Without a dramatic improvement in this environment, we are faced with the very real risk that application developers will reject the exascale systems in favor of smaller, more reliable systems that provide a better overall throughput.
5. *Hierarchical Software Engineering.* The software engineering community, by most accounts, has been slow to adopt formal software engineering practices. Although a lot of high-quality software has been developed without formal practices, the demands of collaborative development, multicode environments, and large collective teams require more attention to the benefits that formal practices can provide. Typically, software engineers work in a tightknit team to rapidly produce small software. However, the newer paradigms demand development of complex and multifold larger software, such as Trilinos, which is an example of a "project of projects." The basic idea undertaken when developing Trilinos was (1) cross-fertilization of ideas, techniques, and tools across package teams; (2) adoption of "best practices" from one package across other packages; (3) fostering of trust among disparate groups; (4) software modularity that is naturally enforced by package and team boundaries; and (5) well-defined interfaces between packages for interoperability. The aforementioned may lead to better large-scale software development, but of course, a serious research effort must be undertaken in this direction.

REFERENCES

[1] L. Kuan-Ching, H. Ching-Hsien, and T.Y. Laurence, *Handbook of Research on Scalable Computing Technologies*. Hershey, PA: IGI Global snippet, 2009.

[2] Y.C. Kwong, *Annual Review of Scalable Computing*. Singapore: World Scientific, 2004.

[3] L. Valentini, W. Lassonde, S.U. Khan, N. Min-Allah, S.A. Madani, J. Li, L. Zhang, L. Wang, N. Ghani, J. Kolodziej, H. Li, A.Y. Zomaya, C. Xu, P. Balaji, A. Vishnu, F. Pinel, J.E. Pecero, D. Kliazovich, and P. Bouvry, "An overview of energy efficiency techniques in cluster computing systems," *Cluster Computing*, forthcoming. DOI:10.1007/s10586-011-0171-x.

[4] R. Buyya, *High Performance Cluster Computing: Systems and Architectures*. Prentice Hall, 1999.

[5] J. Joseph and C. Fellenstein, *Grid Computing*. Upper Saddle River, NJ: Prentice Hall, 2005.

[6] L. Wang, W. Jie, and J. Chen, *Grid Computing: Infrastructure, Service, and Applications*. Boca Raton, FL: CRC Press, 2009.

[7] W. Peng, J. Tao, L. Wang, H. Marten, and D. Chen, "Towards providing cloud functionalities for grid users," *IEEE 17th International Conference on Parallel and Distributed Systems (ICPADS)*, December 2011, Tainan, Taiwan.

[8] J. Kolodziej and S.U. Khan, "Data scheduling in data grids and data centers: A short taxonomy of problems and intelligent resolution techniques," *LNCS Transactions on Computational Collective Intelligence*, forthcoming.

[9] P. Lindberg, J. Leingang, D. Lysaker, K. Bilal, S.U. Khan, P. Bouvry, N. Ghani, N. Min-Allah, and J. Li, "Comparison and analysis of greedy energy-efficient scheduling algorithms for computational grids," in *Energy Aware Distributed Computing Systems* (A.Y. Zomaya and Y.C. Lee, eds.), forthcoming.

[10] D. Kliazovich, P. Bouvry, and S.U. Khan, "GreenCloud: A packet-level simulator of energy-aware cloud computing data centers," *Journal of Supercomputing*, forthcoming. DOI: 10.1007/s11227-010-0504-1.

[11] G.L. Valentini, S.U. Khan, and P. Bouvry, "Energy-efficient resource utilization in cloud computing," in *Large Scale Network-centric Computing Systems* (A.Y. Zomaya and A.H. Sarbazi-Azad, eds.), forthcoming.

[12] J. Li, Q. Li, S.U. Khan, and N. Ghani, "Community-based cloud for emergency management," *6th IEEE International Conference on System of Systems Engineering (SoSE)*, June 2011, Albuquerque, NM.

[13] D. Kliazovich, P. Bouvry, Y. Audzevich, and S.U. Khan, "GreenCloud: A packet-level simulator of energy-aware cloud computing data centers," *53rd IEEE Global Communications Conference (Globecom)*, December 2010, Miami, FL.

[14] T. Velte, A.T. Velte, T.J. Velte, and R.C. Elsenpeter, *Cloud Computing, a Practical Approach*. New York: McGraw-Hill, 2009.

[15] N. Antonopoulos and L. Gillam, *Cloud Computing: Principles, Systems and Applications*. London: Springer, 2010.

[16] R. Buyya, J. Broberg, and A.M. Goscinski, *Cloud Computing: Principles and Paradigms*. Hoboken, NJ: Wiley, 2011.

[17] B. Sosinsky, *Cloud Computing Bible*. Hoboken, NJ: Wiley, 2011.

[18] D. Kliazovich, P. Bouvry, and S.U. Khan, "DENS: Data center energy-efficient network-aware scheduling," *Cluster Computing*, forthcoming. DOI: 10.1007/s10586-011-0177-4.

[19] L. Wang and S.U. Khan, "Review of performance metrics for Green data centers: A taxonomy study," *Journal of Supercomputing*, forthcoming. DOI: 10.1007/s11227-011-0704-3.

[20] L. Wang, S.U. Khan, and J. Dayal, "Thermal aware workload placement with task-temperature profiles in a data center," *Journal of Supercomputing*, forthcoming. DOI: 10.1007/s11227-011-0635-z.

[21] S.U. Khan and N. Min-Allah, "A goal programming based energy efficient resource allocation in data centers," *Journal of Supercomputing*, forthcoming. DOI: 10.1007/s11227-011-0611-7.

[22] K. Bilal, S.U. Khan, J. Kolodziej, L. Zhang, K. Hayat, S.A. Madani, N. Min-Allah, L. Wang, and D. Chen, "A comparative study of data center network architectures," *26th European Conference on Modeling and Simulation (ECMS)*, May 2012, Koblenz, Germany.

[23] S.U. Khan and C. Ardil, "A weighted sum technique for the joint optimization of performance and power consumption in data centers," *International Journal of Electrical, Computer, and Systems Engineering*, 3(1):35–40, 2009.

[24] S.U. Khan, "A multi-objective programming approach for resource allocation in data centers," *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, July 2009, Las Vegas, NV.

[25] S.U. Khan, "A self-adaptive weighted sum technique for the joint optimization of performance and power consumption in data centers," *22nd International Conference on Parallel and Distributed Computing and Communication Systems (PDCCS)*, September 2009, Louisville, KY.

[26] L. Wang, L. Von, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, "Cloud computing: A perspective study," *New Generation Computing*, 28(2):137–146, 2010.

[27] L. Wang, R. Ranjan, J. Chen, and B. Benatallah, *Cloud Computing: Methodology, Systems, and Applications*. Boca Raton, FL: Taylor & Francis, 2011.

[28] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," *Journal of Internet Services and Applications*, 1(1):7–18, 2010.

[29] A. Beloglazov, R. Buyya, Y.L. Choon, and A. Zomaya, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," in *Advances in Computers* (M. Zelkowitz, ed.), San Diego, CA: Elsevier Science, 2011.

[30] IEEE P2301. Available at http://standards.ieee.org/develop/wg/CPWG-2301_WG.html. Accessed April 12, 2012.

[31] IEEE P2302. Available at http://standards.ieee.org/develop/wg/ICWG-2302_WG.html. Accessed Apr 12, 2012.

[32] L. Wang, J. Tao, H. Marten, A. Streit, S.U. Khan, J. Kolodziej, and D. Chen, "MapReduce across distributed clusters for data-intensive applications," *26th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2012, Shanghai, China.

[33] L. Perkins, P. Andrews, D. Panda, D. Morton, R. Bonica, N. Werstiuk, and R. Kreiser, "Data intensive computing," *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, 2006, Tampa, FL.

[34] I. Gorton, P. Greenfield, A. Szalay, and R. Williams, "Data-intensive computing in the 21st century," *Computers*, 44(4):30–32, 2008.

[35] M.D. McCool, "Scalable programming models for massively multicore processors," *Proceedings of the IEEE*, 96(5):816–831, 2008.

[36] K. Hwang and Z. Xu, *Scalable Parallel Computing: Technology, Architecture, Programming*. New York: McGraw-Hill, 1998.

[37] S. Ali, A. Maciejewski, H.J. Siegel, and J. Kim, "Measuring the robustness of a resource allocation," *IEEE Transactions on Parallel and Distributed Systems*, 15(7):630–641, 2004.