Big Data Processing in Cloud Computing Environments

Changqing Ji*†, Yu Li‡, Wenming Qiu‡, Uchechukwu Awada‡, Keqiu Li‡
*College of Information Science and Technology, Dalian Maritime University, Dalian 116026, China
†College of Physical Science and Technology, Dalian University, Dalian 116600, China
‡School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

Email:{jcqgood, liyu87122, xmdlut2007, awadauche, likeqiu}@gmail.com

***Abstract*—With the rapid growth of emerging applications like social network analysis, semantic Web analysis and bioinformatics network analysis, a variety of data to be processed continues to witness a quick increase. Effective management and analysis of large-scale data poses an interesting but critical challenge. Recently, big data has attracted a lot of attention from academia, industry as well as government. This paper introduces several big data processing technics from system and application aspects. First, from the view of cloud data management and big data processing mechanisms, we present the key issues of big data processing, including cloud computing platform, cloud architecture, cloud database and data storage scheme. Following the MapReduce parallel processing framework, we then introduce MapReduce optimization strategies and applications reported in the literature. Finally, we discuss the open issues and challenges, and deeply explore the research directions in the future on big data processing in cloud computing environments.**

***Keywords*-Big Data; Cloud Computing; Data Management; Distributed Computing.**

I. Introduction

In the last two decades, the continuous increase of computational power has produced an overwhelming flow of data. Big data is not only becoming more available but also more understandable to computers. For example, modern high-energy physics experiments, such as DZero[1], typically generate more than one TeraByte of data per day. The famous social network Website, Facebook, serves 570 billion page views per month, stores 3 billion new photos every month, and manages 25 billion pieces of content[2]. Google's search and ad business, Facebook, Flickr, YouTube, and Linkedin use a bundle of artificial-intelligence tricks, require parsing vast quantities of data and making decisions instantaneously. Multimedia data mining platforms make it easy for everybody to achieve these goals with the minimum amount of effort in terms of software, CPU and network. On March 29, 2012, American government announced the "Big Data Research and Development Initiative", and big data becomes the national policy for the first time[3]. All these examples showed that daunting big data challenges and significant resources were allocated to support these data-intensive operations which lead to high storage and data processing costs.

The current technologies such as grid and cloud computing have all intended to access large amounts of computing power by aggregating resources and offering a single system view. Among these technologies, cloud computing is becoming a powerful architecture to perform large-scale and complex computing, and has revolutionized the way that computing infrastructure is abstracted and used. In addition, an important aim of these technologies is to deliver computing as a solution for tackling big data, such as large-scale, multi-media and high dimensional data sets.

Big data and cloud computing are both the fastest-moving technologies identified in Gartner Inc.'s 2012 Hype Cycle for Emerging Technologies[4]. Cloud computing is associated with new paradigm for the provision of computing infrastructure and big data processing method for all kinds of resources. Moreover, some new cloud-based technologies have to be adopted because dealing with big data for concurrent processing is difficult.

Then what is Big Data? In the publication of the journal of Science 2008, "Big Data" is defined as "Represents the progress of the human cognitive processes, usually includes data sets with sizes beyond the ability of current technology, method and theory to capture, manage, and process the data within a tolerable elapsed time"[1]. Recently, the definition of big data as also given by the Gartner: "Big Data are high-volume, high-velocity, and/or high-variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization"[2]. According to Wikimedia, "In information technology, big data is a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools"[5].

The goal of this paper is to provide the status of big data studies and related works, which aims at providing a general view of big data management technologies and

[1]http://www-d0.fnal.gov/
[2]http://www.facebook.com
[3]http://www.whitehouse.gov/blog/2012/03/29/big-data-big-deal
[4]http://www.gartner.com
[5]http://en.wikipedia.org/wiki/Big-data

1087-4089/12 $26.00 © 2012 IEEE
DOI 10.1109/I-SPAN.2012.9

applications. We give an overview of major approaches and classify them with respect to their strategies including big data management platform, distributed file system, big data storage, MapReduce application and optimization. However, maintaining and processing these large-scale data sets is typically beyond the reach of small businesses and it is increasingly posing challenges even for large companies and institutes. Finally, we discuss the open issues and challenges in processing big data in three important aspects: big data storage, analysis and security.

The rest of the paper is organized as follows. Section 2 reviews the architecture and the key concepts of big data processing. Section 3 presents the classification of major applications and optimization of the MapReduce framework while Section 4 discusses several open issues and future challenges. Finally, Section 5 concludes this paper.

II. Big Data Management System

Many researchers have suggested that commercial DBMSs are not suitable for processing extremely large scale data. Classic architecture's potential bottleneck is the database server while faced with peak workloads. One database server has restriction of scalability and cost, which are two important goals of big data processing. In order to adapt various large data processing models, D. Kossmann et al. presented four different architectures based on classic multi-tier database application architecture which are partitioning, replication, distributed control and caching architecture[3]. It is clear that the alternative providers have different business models and target different kinds of applications: Google seems to be more interested in small applications with light workloads whereas Azure is currently the most affordable service for medium to large services. Most of recent cloud service providers are utilizing hybrid architecture that is capable of satisfying their actual service requirements. In this section, we mainly discuss big data architecture from three key aspects: distributed file system, non-structural and semi-structured data storage and open source cloud platform.

A. Distributed File System

Google File System (GFS)[4] is a chunk-based distributed file system that supports fault-tolerance by data partitioning and replication. As an underlying storage layer of Google's cloud computing platform, it is used to read input and store output of MapReduce[5]. Similarly, Hadoop also has a distributed file system as its data storage layer called Hadoop Distributed File System (HDFS)[6], which is an open-source counterpart of GFS. GFS and HDFS are user-level filesystems that do not implement POSIX semantics and heavily optimized for the case of large files (measured in gigabytes)[7]. Amazon Simple Storage Service (S3)[8] is an online public storage web service offered by Amazon Web Services. This file system is targeted at clusters hosted on the Amazon Elastic Compute Cloud server-on-demand infrastructure. S3 aims to provide scalability, high availability, and low latency at commodity costs. ES2[9] is an elastic storage system of epiC[6], which is designed to support both functionalities within the same storage. The system provides efficient data loading from different sources, flexible data partitioning scheme, index and parallel sequential scan. In addition, there are general filesystems that have not to be addressed such as Moose File System (MFS)[7], Kosmos Distributed Filesystem (KFS)[8].

B. Non-structural and Semi-structured Data Storage

With the success of the Web 2.0, more and more IT companies have increasing needs to store and analyze the ever growing data, such as search logs, crawled web content, and click streams, usually in the range of petabytes, collected from a variety of web services. However, web data sets are usually non-relational or less structured and processing such semi-structured data sets at scale poses another challenge. Moreover, simple distributed file systems mentioned above cannot satisfy service providers like Google, Yahoo!, Microsoft and Amazon. All providers have their purpose to serve potential users and own their relevant state-of-the-art of big data management systems in the cloud environments. Bigtable[10] is a distributed storage system of Google for managing structured data that is designed to scale to a very large size (petabytes of data) across thousands of commodity servers. Bigtable does not support a full relational data model. However, it provides clients with a simple data model that supports dynamic control over data layout and format. PNUTS[11] is a massive-scale hosted database system designed to support Yahoo!'s web applications. The main focus of the system is on data serving for web applications, rather than complex queries. Upon PNUTS, new applications can be built very easily and the overhead of creating and maintaining these applications is nothing much. The Dynamo[12] is a highly available and scalable distributed key/value based data store built for supporting internal Amazon's applications. It provides a simple primary-key only interface to meet the requirements of these applications. However, it differs from key-value storage system. Facebook proposed the design of a new cluster-based data warehouse system, Llama[13], a hybrid data management system which combines the features of row-wise and column-wise database systems. They also describe a new column-wise file format for Hadoop called CFile, which provides better performance than other file formats in data analysis.

[6]http://www.comp.nus.edu.sg/ epic/overview.html
[7]http://www.moosefs.org/
[8]http://kosmosfs.sourceforge.net/

C. Open Source Cloud Platform

The main idea behind data center is to leverage the virtualization technology to maximize the utilization of computing resources. Therefore, it provides the basic ingredients such as storage, CPUs, and network bandwidth as a commodity by specialized service providers at low unit cost. For reaching the goals of big data management, most of the research institutions and enterprises bring virtualization into cloud architectures. Amazon Web Services (AWS), Eucalptus, Opennebula, Cloudstack and Openstack are the most popular cloud management platforms for infrastructure as a service (IaaS). AWS[9] is not free but it has huge usage in elastic platform. It is very easy to use and only pay-as-you-go. The Eucalyptus[14] works in IaaS as an open source. It uses virtual machine in controlling and managing resources. Since Eucalyptus is the earliest cloud management platform for IaaS, it signs API compatible agreement with AWS. It has a leading position in the private cloud market for the AWS ecological environment. OpenNebula[15] has integration with various environments. It can offer the richest features, flexible ways and better interoperability to build private, public or hybrid clouds. OpenNebula is not a Service Oriented Architecture (SOA) design and has weak decoupling for computing, storage and network independent components. CloudStack[10] is an open source cloud operating system which delivers public cloud computing similar to Amazon EC2 but using users' own hardware. CloudStack users can take full advantage of cloud computing to deliver higher efficiency, limitless scale and faster deployment of new services and systems to the end-user. At present, CloudStack is one of the Apache open source projects. It already has mature functions. However, it needs to further strengthen the loosely coupling and component design. OpenStack[11] is a collection of open source software projects aiming to build an open-source community with researchers, developers and enterprises. People in this community share a common goal to create a cloud that is simple to deploy, massively scalable and full of rich features. The architecture and components of OpenStack are straightforward and stable, so it is a good choice to provide specific applications for enterprises. In current situation, OpenStack has good community and ecological environment. However, it still have some shortcomings like incomplete functions and lack of commercial supports.

III. Applications and Optimization

A. Application

In this age of data explosion, parallel processing is essential to perform a massive volume of data in a timely manner. The use of parallelization techniques and algorithms is the key to achieve better scalability and performance for processing big data. At present, there are a lot of popular parallel processing models, including MPI, General Purpose GPU (GPGPU), MapReduce and MapReduce-like. MapReduce proposed by Google, is a very popular big data processing model that has rapidly been studied and applied by both industry and academia. MapReduce has two major advantages: the MapReduce model hide details related to the data storage, distribution, replication, load balancing and so on. Furthermore, it is so simple that programmers only specify two functions, which are map function and reduce function, for performing the processing of the big data. We divided existing MapReduce applications into three categories: partitioning sub-space, decomposing sub-processes and approximate overlapping calculations.

While MapReduce is referred to as a new approach of processing big data in cloud computing environments, it is also criticized as a "major step backwards" compared with DBMS[16]. We all know that MapReduce is schema-free and index-free. Thus, the MapReduce framework requires parsing each record at reading input. As the debate continues, the final result shows that neither is good at the other does well, and the two technologies are complementary[17]. Recently, some DBMS vendors also have integrated MapReduce front-ends into their systems including Aster, HadoopDB[18], Greenplum[19] and Vertuca. Mostly of those are still databases, which simply provide a MapReduce front-end to a DBMS. HadoopDB is a hybrid system which efficiently takes the best features from the scalability of MapReduce and the performance of DBMS. The result shows that HadoopDB improves task processing times of Hadoop by a large factor to match the shared-nothing DBMS. Lately, J. Dittrich et al. propose a new type of system named Hadoop++[20] which indicates that HadoopDB has also severe drawbacks, including forcing user to use DBMS, changing the interface to SQL and so on. There are also certain papers adapting different inverted index, which is a simple but practical index structure and appropriate for MapReduce to process big data, such as [21] etc. We also do intensive study on large-scale spatial data environment and design a distributed inverted grid index by combining inverted index and spatial grid partition with MapReduce model, which is simple, dynamic, scale and fit for processing high dimensional spatial data[22].

MapReduce has received a lot of attentions in many fields, including data mining, information retrieval, image retrieval, machine learning, and pattern recognition. For example, Mahout[12] is an Apache project that aims at building scalable machine learning libraries which are all implemented on the Hadoop. However, as the amount of data that need to be processed grows, many data processing methods have

[9] http://aws.amazon.com/what-is-aws/
[10] http://cloudstack.org/software.pdf
[11] http://www.openstack.org/
[12] http://mahout.apache.org/

become not suitable or limited. Recently, many research efforts have exploited the MapReduce framework for solving challenging data processing problems on large scale datasets in different domains. For example, the Ricardo[23] is soft system that integrate *R statistical tool* and Hadoop to support parallel data analysis. RankReduce[24] perfectly combines the Local Sensitive Hashing (LSH) and MapReduce, which effectively performs K-Nearest Neighbors search in the high-dimensional spaces. F. Cordeiro et al.[25] proposed BoW method for clustering very large and multi-dimensional datasets with MapReduce which is a hard-clustering method and allows the automatic, and dynamic trade-off between disk delay and network delay. MapDupReducer[26] is a MapReduce based system capable of detecting near duplicates over massive datasets efficiently. In addition, C. Ranger et al.[27] implement the MapReduce framework on multiple processors in a single machine, which has gained good performance. Recently, B. He et al. develop Mars[28], a GPU-based MapReduce framework, which gains better performance than the state-of-the-art CPU-based framework.

B. Optimization

In this section, we present details of approaches to improve the performance of processing big data with MapReduce.

1) Data Transfer Bottlenecks: It is a big challenge that cloud users must consider how to minimize the cost of data transmission. Consequently, researchers have begun to propose variety of approaches. Map-Reduce-Merge[29] is a new model that adds a Merge phase after Reduce phase that combines two reduced outputs from two different MapReduce jobs into one, which can efficiently merge data that is already partitioned and sorted (or hashed) by map and reduce modules. Map-Join-Reduce[30] is a system that extends and improves MapReduce runtime framework by adding Join stage before Reduce stage to perform complex data analysis tasks on large clusters. They present a new data processing strategy which runs filtering-join-aggregation tasks with two consecutive MR jobs. It adopts one-to-many shuffling scheme to avoid frequent checkpointing and shuffling of intermediate results. Moreover, different jobs often perform similar work, thus sharing similar work reduces overall amount of data transfer between jobs. MRShare[31] is a sharing framework proposed by T. Nykiel et al. that transforms a batch of queries into a new batch that can be executed more efficiently by merging jobs into groups and evaluating each group as a single query. Data skew is also an important factor that affects data transfer cost. In order to overcome this deficiency, we propose a method[32] that divides a MapReduce job into two phases: sampling MapReduce job and expected MapReduce job. The first phase is to sample the input data, gather the inherent distribution on keys' frequencies and then make a good partition scheme in advance. In the second phase, expected MapReduce job applies this partition scheme to every mapper to group the intermediate keys quickly.

2) Iterative Optimization: MapReduce also is a popular platform in which the dataflow takes the form of a directed acyclic graph of operators. However, it requires lots of I/Os and unnecessary computations while solving the problem of iterations with MapReduce. Twister[33] proposed by J. Ekanayake et al. is an enhanced MapReduce runtime that supports iterative MapReduce computations efficiently, which adds an extra Combine stage after Reduce stage. Thus, data output from combine stage flows to the next iteration's Map stage. It avoids instantiating workers repeatedly during iterations and previously instantiated workers are reused for the next iteration with different inputs. HaLoop[34] is similar to Twister, which is a modified version of the MapReduce framework that supports for iterative applications by adding a Loop control. It also allows to cache both stages' input and output to save more I/Os during iterations. There exist lots of iterations during graph data processing. Pregel[35] implements a programming model motivated by the Bulk Synchronous Parallel(BSP) model, in which each node has its own input and transfers only some messages which are required for the next iteration to other nodes.

3) Online: There are some jobs which need to process online while original MapReduce can not do this very well. MapReduce Online[36] is desgined to support online aggregation and continuous queries in MapReduce. It raises an issue that frequent checkpointing and shuffling of intermediate results limit pipelined processing. They modify MapReduce framework by making Mappers push their data temporarily stored in local storage to Reducers preiodically in the same MR job. In addition, Map-side pre-aggregation is used to reduce communication. Hadoop Online Prototype (HOP)[37] proposed by Tyson Condie is similar to MapReduce Online. HOP is a modified version of MapReduce framework that allows users to early get returns from a job as it is being computed. It also supports for continuous queries which enable MapReduce programs to be written for applications such as event monitoring and stream processing while retaining the fault tolerance properties of Hadoop. D. Jiang et al.[38] found that the merge sort in MapReduce costs lots of I/Os and seriously affects the performance of MapReduce. In the study, the results are hashed and pushed to hash tables held by reducers as soon as each map task outputs its intermediate results. Then, reducers perform aggregation on the values in each bucket. Since each bucket in the hash table holds all values which correspond to a distinct key, no grouping is required. In addition, reducers can perform aggregation on the fly even when all mappers are not completed yet.

4) Join Query Optimization: Join Query is a popular problem in big data area. However a join problem needs

more than two inputs while MapReduce is devised for processing a single input. R. Vernica et al.[39] proposed a 3-stage approach for end-to-end set-similarity joins. They efficiently partition the data across nodes in order to balance the workload and minimize the need for replication. Wei Lu et al. investigate how to perform kNN join using MapReduce[40]. Mappers cluster objects into groups, then Reducers perform the kNN join on each group of objects separately. To reduce shuffling and computational costs, they desgine an effective mapping mechanism that exploits pruning rules for distance filtering. In addition, two approximate algorithms minimize the number of replicas to reduce the shuffling cost.

IV. Discussion and Challenges

We are now in the days of big data. We can gather more information from daily life of every human being. The top seven big data drivers are science data, Internet data, finance data, mobile device data, sensor data, RFID data and streaming data. Coupled with recent advances in machine learning and reasoning, as well as rapid rises in computing power and storage, we are transforming our ability to make sense of these increasingly large, heterogeneous, noisy and incomplete datasets collected from a variety of sources.

So far, researchers are not able to unify around the essential features of big data. Some think that big data is the data that we are not able to process using pre-exist technology, method and theory. However, no matter how we consider the definition of big data, the world is turning into a ”helplessness” age while varies of incalculable data is being generated by science, business and society. Big data put forward new challenges for data management and analysis, and even for the whole IT industry.

We consider there are three important aspects while we encounter with problems in processing big data, and we present our points of view in details as follows.

Big Data Storage and Management: Current technologies of data management systems are not able to satisfy the needs of big data, and the increasing speed of storage capacity is much less than that of data, thus a revolution re-construction of information framework is desperately needed. We need to design a hierarchical storage architecture. Besides, previous computer algorithms are not able to effectively storage data that is directly acquired from the actual world, due to the heterogeneity of the big data. However, they perform excellent in processing homogeneous data. Therefore, how to re-organize data is one big problem in big data management. Virtual server technology can exacerbate the problem, raising the prospect of overcommitted resources, especially if communication is poor between the application, server and storage administrators. We also need to solve the bottleneck problems of the high concurrent I/O and single-named node in the present Master-Slave system model.

Big Data Computation and Analysis: While processing a query in big data, speed is a significant demand[41]. However, the process may take time because mostly it cannot traverse all the related data in the whole database in a short time. In this case, index will be an optimal choice. At present, indices in big data are only aiming at simple type of data, while big data is becoming more complicated. The combination of appropriate index for big data and up-to-date preprocessing technology will be a desirable solution when we encountered this kind of problems. Application parallelization and divide-and-conquer is natural computational paradigms for approaching big data problems. But getting additional computational resources is not as simple as just upgrading to a bigger and more powerful machine on the fly. The traditional serial algorithm is inefficient for the big data. If there is enough data parallelism in the application, users can take advantage of the cloud’s reduced cost model to use hundreds of computers for a short time costs.

Big Data Security: By using online big data application, a lot of companies can greatly reduce their IT cost. However, security and privacy affect the entire big data storage and processing, since there is a massive use of third-party services and infrastructures that are used to host important data or to perform critical operations. The scale of data and applications grow exponentially, and bring huge challenges of dynamic data monitoring and security protection. Unlike traditional security method, security in big data is mainly in the form of how to process data mining without exposing sensitive information of users. Besides, current technologies of privacy protection are mainly based on static data set, while data is always dynamically changed, including data pattern, variation of attribute and addition of new data. Thus, it is a challenge to implement effective privacy protection in this complex circumstance. In addition, legal and regulatory issues also need attention.

V. Conclusion

This paper described a systematic flow of survey on the big data processing in the context of cloud computing. We respectively discussed the key issues, including cloud storage and computing architecture, popular parallel processing framework, major applications and optimization of MapReduce. Big Data is not a new concept but very challenging. It calls for scalable storage index and a distributed approach to retrieve required results near real-time. It is a fundamental fact that data is too big to process conventionally. Nevertheless, big data will be complex and exist continuously during all big challenges, which are the big opportunities for us. In the future, significant challenges need to be tackled by industry and academia. It is an urgent need that computer scholars and social sciences scholars make close cooperation, in order to guarantee the long-term success of cloud computing and collectively explore new

territory.

REFERENCES

[1] "Big data: science in the petabyte era," *Nature 455 (7209): 1*, 2008.

[2] Douglas and Laney, "The importance of 'big data': A definition," 2008.

[3] D. Kossmann, T. Kraska, and S. Loesing, "An evaluation of alternative architectures for transaction processing in the cloud," in *Proceedings of the 2010 international conference on Management of data*. ACM, 2010, pp. 579–590.

[4] S. Ghemawat, H. Gobioff, and S. Leung, "The google file system," in *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5. ACM, 2003, pp. 29–43.

[5] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[6] D. Borthakur, "The hadoop distributed file system: Architecture and design," *Hadoop Project Website*, vol. 11, 2007.

[7] A. Rabkin and R. Katz, "Chukwa: A system for reliable large-scale log collection," in *USENIX Conference on Large Installation System Administration*, 2010, pp. 1–15.

[8] S. Sakr, A. Liu, D. Batista, and M. Alomari, "A survey of large scale data management approaches in cloud environments," *Communications Surveys & Tutorials, IEEE*, vol. 13, no. 3, pp. 311–336, 2011.

[9] Y. Cao, C. Chen, F. Guo, D. Jiang, Y. Lin, B. Ooi, H. Vo, S. Wu, and Q. Xu, "Es2: A cloud data storage system for supporting both oltp and olap," in *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*. IEEE, 2011, pp. 291–302.

[10] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. Gruber, "Bigtable: A distributed structured data storage system," in *7th OSDI*, 2006, pp. 305–314.

[11] B. Cooper, R. Ramakrishnan, U. Srivastava, A. Silberstein, P. Bohannon, H. Jacobsen, N. Puz, D. Weaver, and R. Yerneni, "Pnuts: Yahoo!'s hosted data serving platform," *Proceedings of the VLDB Endowment*, vol. 1, no. 2, pp. 1277–1288, 2008.

[12] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Dynamo: amazon's highly available key-value store," in *ACM SIGOPS Operating Systems Review*, vol. 41, no. 6. ACM, 2007, pp. 205–220.

[13] Y. Lin, D. Agrawal, C. Chen, B. Ooi, and S. Wu, "Llama: leveraging columnar storage for scalable join processing in the mapreduce framework," in *Proceedings of the 2011 international conference on Management of data*. ACM, 2011, pp. 961–972.

[14] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The eucalyptus open-source cloud-computing system," in *Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on*. IEEE, 2009, pp. 124–131.

[15] P. Sempolinski and D. Thain, "A comparison and critique of eucalyptus, opennebula and nimbus," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*. Ieee, 2010, pp. 417–426.

[16] D. DeWitt and M. Stonebraker, "Mapreduce: A major step backwards," *The Database Column*, vol. 1, 2008.

[17] M. Stonebraker, D. Abadi, D. DeWitt, S. Madden, E. Paulson, A. Pavlo, and A. Rasin, "Mapreduce and parallel dbmss: friends or foes," *Communications of the ACM*, vol. 53, no. 1, pp. 64–71, 2010.

[18] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin, "Hadoopdb: an architectural hybrid of mapreduce and dbms technologies for analytical workloads," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 922–933, 2009.

[19] Y. Xu, P. Kostamaa, and L. Gao, "Integrating hadoop and parallel dbms," in *Proceedings of the 2010 international conference on Management of data*. ACM, 2010, pp. 969–974.

[20] J. Dittrich, J. Quiané-Ruiz, A. Jindal, Y. Kargin, V. Setty, and J. Schad, "Hadoop++: Making a yellow elephant run like a cheetah (without it even noticing)," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 515–529, 2010.

[21] D. Logothetis and K. Yocum, "Ad-hoc data processing in the cloud," *Proceedings of the VLDB Endowment*, vol. 1, no. 2, pp. 1472–1475, 2008.

[22] C. Ji, T. Dong, Y. Li, Y. Shen, K. Li, W. Qiu, W. Qu, and M. Guo, "Inverted grid-based knn query processing with mapreduce," in *ChinaGrid, 2012 Seventh ChinaGrid Annual Conference on*. IEEE, 2012, pp. 25–33.

[23] S. Das, Y. Sismanis, K. Beyer, R. Gemulla, P. Haas, and J. McPherson, "Ricardo: integrating r and hadoop," in *Proceedings of the 2010 international conference on Management of data*. ACM, 2010, pp. 987–998.

[24] A. Stupar, S. Michel, and R. Schenkel, "Rankreduce–processing k-nearest neighbor queries on top of mapreduce," in *Proceedings of the 8th Workshop on Large-Scale Distributed Systems for Information Retrieval*, 2010, pp. 13–18.

[25] R. Ferreira Cordeiro, C. Traina Junior, A. Machado Traina, J. López, U. Kang, and C. Faloutsos, "Clustering very large multi-dimensional datasets with mapreduce," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 690–698.

[26] C. Wang, J. Wang, X. Lin, W. Wang, H. Wang, H. Li, W. Tian, J. Xu, and R. Li, "Mapdupreducer: detecting near duplicates over massive datasets," in *Proceedings of the 2010 international conference on Management of data*. ACM, 2010, pp. 1119–1122.

[27] C. Ranger, R. Raghuraman, A. Penmetsa, G. Bradski, and C. Kozyrakis, "Evaluating mapreduce for multi-core and multiprocessor systems," in *High Performance Computer Architecture, 2007. HPCA 2007. IEEE 13th International Symposium on*. IEEE, 2007, pp. 13–24.

[28] B. He, W. Fang, Q. Luo, N. Govindaraju, and T. Wang, "Mars: a mapreduce framework on graphics processors," in *Proceedings of the 17th international conference on Parallel architectures and compilation techniques*. ACM, 2008, pp. 260–269.

[29] H. Yang, A. Dasdan, R. Hsiao, and D. Parker, "Map-reduce-merge: simplified relational data processing on large clusters," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM, 2007, pp. 1029–1040.

[30] D. Jiang, A. Tung, and G. Chen, "Map-Join-Reduce: Toward scalable and efficient data analysis on large clusters," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 23, no. 9, pp. 1299–1311, 2011.

[31] T. Nykiel, M. Potamias, C. Mishra, G. Kollios, and N. Koudas, "Mrshare: Sharing across multiple queries in mapreduce," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 494–505, 2010.

[32] Y. Xu, P. Zou, W. Qu, Z. Li, K. Li, and X. Cui, "Sampling-based partitioning in mapreduce for skewed data," in *China-*

Grid, 2012 Seventh ChinaGrid Annual Conference on. IEEE, 2012, pp. 1–8.

[33] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S. Bae, J. Qiu, and G. Fox, "Twister: a runtime for iterative mapreduce," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. ACM, 2010, pp. 810–818.

[34] Y. Bu, B. Howe, M. Balazinska, and M. Ernst, "Haloop: Efficient iterative data processing on large clusters," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 285–296, 2010.

[35] G. Malewicz, M. Austern, A. Bik, J. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: a system for large-scale graph processing," in *Proceedings of the 2010 international conference on Management of data*. ACM, 2010, pp. 135–146.

[36] T. Condie, N. Conway, P. Alvaro, J. Hellerstein, K. Elmeleegy, and R. Sears, "Mapreduce online," in *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, 2010, pp. 21–21.

[37] T. Condie, N. Conway, P. Alvaro, J. Hellerstein, J. Gerth, J. Talbot, K. Elmeleegy, and R. Sears, "Online aggregation and continuous query support in mapreduce," in *ACM SIGMOD*, 2010, pp. 1115–1118.

[38] D. Jiang, B. Ooi, L. Shi, and S. Wu, "The performance of mapreduce: An in-depth study," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 472–483, 2010.

[39] R. Vernica, M. Carey, and C. Li, "Efficient parallel set-similarity joins using mapreduce," in *SIGMOD conference*. Citeseer, 2010, pp. 495–506.

[40] C. Zhang, F. Li, and J. Jestes, "Efficient parallel knn joins for large data in mapreduce," in *Proceedings of the 15th International Conference on Extending Database Technology*. ACM, 2012, pp. 38–49.

[41] X. Zhou, J. Lu, C. Li, and X. Du, "Big data challenge in the management perspective," *Communications of the CCF*, vol. 8, pp. 16–20, 2012.