IEEE Access
practical innovations : open solutions

Received April 30, 2014, accepted May 12, 2014, date of publication June 24, 2014, date of current version July 8, 2014.

Digital Object Identifier 10.1109/ACCESS.2014.2332453

Toward Scalable Systems for Big Data Analytics: A Technology Tutorial

HAN HU[1], YONGGANG WEN[2], (Senior Member, IEEE), TAT-SENG CHUA[1], AND XUELONG LI[3], (Fellow, IEEE)

[1]School of Computing, National University of Singapore, Singapore 117417
[2]School of Computer Engineering, Nanyang Technological University, Singapore 639798
[3]State Key Laboratory of Transient Optics and Photonics, Center for Optical Imagery Analysis and Learning, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China

Corresponding author: Y. Wen (ygwen@ntu.edu.sg)

This work was supported in part by the Energy Market Authority of Singapore under Grant NRF2012EWT-EIRP002-013, in part by the Singapore National Research Foundation through the International Research Center at Singapore Funding Initiative and administered by the IDM Programme Office, and in part by the National Natural Science Foundation of China under Grant 61125106.

ABSTRACT Recent technological advancements have led to a deluge of data from distinctive domains (e.g., health care and scientific sensors, user-generated data, Internet and financial companies, and supply chain systems) over the past two decades. The term big data was coined to capture the meaning of this emerging trend. In addition to its sheer volume, big data also exhibits other unique characteristics as compared with traditional data. For instance, big data is commonly unstructured and require more real-time analysis. This development calls for new system architectures for data acquisition, transmission, storage, and large-scale data processing mechanisms. In this paper, we present a literature survey and system tutorial for big data analytics platforms, aiming to provide an overall picture for nonexpert readers and instill a do-it-yourself spirit for advanced audiences to customize their own big-data solutions. First, we present the definition of big data and discuss big data challenges. Next, we present a systematic framework to decompose big data systems into four sequential modules, namely data generation, data acquisition, data storage, and data analytics. These four modules form a big data value chain. Following that, we present a detailed survey of numerous approaches and mechanisms from research and industry communities. In addition, we present the prevalent Hadoop framework for addressing big data challenges. Finally, we outline several evaluation benchmarks and potential research directions for big data systems.

INDEX TERMS Big data analytics, cloud computing, data acquisition, data storage, data analytics, Hadoop.

I. INTRODUCTION

The emerging big-data paradigm, owing to its broader impact, has profoundly transformed our society and will continue to attract diverse attentions from both technological experts and the public in general. It is obvious that we are living a data deluge era, evidenced by the sheer volume of data from a variety of sources and its growing rate of generation. For instance, an IDC report [1] predicts that, from 2005 to 2020, the global data volume will grow by a factor of 300, from 130 exabytes to 40,000 exabytes, representing a double growth every two years. The term of ''big-data'' was coined to capture the profound meaning of this data-explosion trend and indeed the *data* has been touted as the new oil, which is expected to transform our society. For example, a Mckinsey report [2] states that the potential value of global personal location data is estimated to be $100 billion in revenue to service providers over the next ten years and be as much as $700 billion in value to consumer and business end users. The huge potential associated with big-data has led to an emerging research field that has quickly attracted tremendous interest from diverse sectors, for example, industry, government and research community. The broad interest is first exemplified by coverage on both industrial reports [2] and public media (e.g.,the Economist [3], [4], the New York Times [5], and the National Public Radio (NPR) [6], [7]). Government has also played a major role in creating new programs [8] to accelerate the progress of tackling the big-data challenges. Finally, Nature and Science Magazines have published special issues to discuss the big-data phenomenon and its challenges, expanding its impact beyond technological

domains. As a result, this growing interest in big-data from diverse domains demands a clear and intuitive understanding of its definition, evolutionary history, building technologies and potential challenges.

This tutorial paper focuses on scalable big-data systems, which include a set of tools and mechanisms to load, extract, and improve disparate data while leveraging the massively parallel processing power to perform complex transformations and analysis. Owing to the uniqueness of big-data, designing a scalable big-data system faces a series of technical challenges, including:

- First, due to the variety of disparate data sources and the sheer volume, it is difficult to collect and integrate data with scalability from distributed locations. For instance, more than 175 million tweets containing text, image, video, social relationship are generated by millions of accounts distributed globally [9].
- Second, big data systems need to store and manage the gathered massive and heterogeneous datasets, while provide function and performance guarantee, in terms of fast retrieval, scalability, and privacy protection. For example, Facebook needs to store, access, and analyze over 30 pertabytes of user generate data [9].
- Third, big data analytics must effectively mine massive datasets at different levels in realtime or near realtime - including modeling, visualization, prediction, and optimization - such that inherent promises can be revealed to improve decision making and acquire further advantages.

These technological challenges demand an overhauling re-examination of the current data management systems, ranging from their architectural principle to the implementation details. Indeed, many leading industry companies [10] have discarded the transitional solutions to embrace the emerging big data platforms.

However, traditional data management and analysis systems, mainly based on relational database management system (RDBMS), are inadequate in tackling the aforementioned list of big-data challenges. Specifically, the mismatch between the traditional RDBMS and the emerging big-data paradigm falls into the following two aspects, including:

- From the perspective of data structure, RDBMSs can only support structured data, but offer little support for semi-structured or unstructured data.
- From the perspective of scalability, RDBMSs scale up with expensive hardware and cannot scale out with commodity hardware in parallel, which is unsuitable to cope with the ever growing data volume.

To address these challenges, the research community and industry have proposed various solutions for big data systems in an ac-hoc manner. Cloud computing can be deployed as the infrastructure layer for big data systems to meet certain infrastructure requirements, such as cost-effectiveness, elasticity, and the ability to scale up or down. Distributed file systems [11] and NoSQL [12] databases are suitable for persistent storage and the management of massive scheme-free datasets. MapReduce [13], a programming framework, has achieved great success in processing group-aggregation tasks, such as website ranking. Hadoop [14] integrates data storage, data processing, system management, and other modules to form a powerful system-level solution, which is becoming the mainstay in handling big data challenges. We can construct various big data applications based on these innovative technologies and platforms. In light of the proliferation of big-data technologies, a systematic framework should be in order to capture the fast evolution of big-data research and development efforts and put the development in different frontiers in perspective.

FIGURE 1. A modular data center was built at Nanyang Technological University (NTU) for system/testbed research. The testbed hosts 270 servers organized into 10 racks.

In this paper, learning from our first-hand experience of building a big-data solution on our private modular data center testbed (as illustrated in Fig. 1), we strive to offer a systematic tutorial for scalable big-data systems, focusing on the enabling technologies and the architectural principle. It is our humble expectation that the paper can serve as a first stop for domain experts, big-data users and the general audience to look for information and guideline in their specific needs for big-data solutions. For example, the domain experts could follow our guideline to develop their own big-data platform and conduct research in big-data domain; the big-data users can use our framework to evaluate alternative solutions proposed by their vendors; and the general audience can understand the basic of big-data and its impact on their work and life. For such a purpose, we first present a list of alternative definitions of big data, supplemented with the history of big-data and big-data paradigms. Following that, we introduce a generic framework to decompose big data platforms into four components, i.e., data generation, data acquisition, data storage, and data analysis. For each stage, we survey current research and development efforts and provide engineering insights for architectural design. Moving toward a specific solution, we then delve on Hadoop - the de facto choice for big data analysis platform, and provide benchmark results for big-data platforms.

The rest of this paper is organized as follows. In Section II, we present the definition of big data and its brief history, in addition to processing paradigms. Then, in Section III, we introduce the big data value chain (which is composed of four phases), the big data technology map, the layered system architecture and challenges. The next four sections describe

the different big data phases associated with the big data value chain. Specifically, Section IV focuses on big data generation and introduces representative big data sources. Section V discusses big data acquisition and presents data collection, data transmission, and data preprocessing techniques. Section VI investigates big data storage approaches and programming models. Section VII discusses big data analytics, and several applications are discussed in Section VIII. Section IX introduces Hadoop, which is the current mainstay of the big data movement. Section X outlines several benchmarks for evaluating the performance of big data systems. A brief conclusion with recommendations for future studies is presented in Section XI.

II. BIG DATA: DEFINITION, HISTORY AND PARADIGMS

In this section, we first present a list of popular definitions of big data, followed by a brief history of its evolution. This section also discusses two alternative paradigms, streaming processing and batch processing.

A. BIG DATA DEFINITION

Given its current popularity, the definition of big data is rather diverse, and reaching a consensus is difficult. Fundamentally, big data means not only a large volume of data but also other features that differentiate it from the concepts of ''massive data'' and ''very large data''. In fact, several definitions for big data are found in the literature, and three types of definitions play an important role in shaping how big data is viewed:

- *Attributive Definition*: IDC is a pioneer in studying big data and its impact. It defines big data in a 2011 report that was sponsored by EMC (the cloud computing leader) [15]: ''Big data technologies describe a new generation of technologies and architectures, designed to economically extract value from very large volumes of a wide variety of data, by enabling high-velocity capture, discovery, and/or analysis.'' This definition delineates the four salient features of big data, i.e., volume, variety, velocity and value. As a result, the ''4Vs'' definition has been used widely to characterize big data. A similar description appeared in a 2001 research report [2] in which META group (now Gartner) analyst Doug Laney noted that data growth challenges and opportunities are three-dimensional, i.e., increasing volume, velocity, and variety. Although this description was not meant originally to define big data, Gartner and much of the industry, including IBM [16] and certain Microsoft researchers [17], continue to use this ''3Vs'' model to describe big data 10 years later [18].
- *Comparative Definition*: In 2011, Mckinsey's report [2] defined *big data* as ''datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze.'' This definition is subjective and does not define big data in terms of any particular metric. However, it incorporates an evolutionary aspect in the definition (over time or across sectors) of what a dataset must be to be considered as big data.
- *Architectural Definition*: The National Institute of Standards and Technology (NIST) [19] suggests that, ''Big data is where the data volume, acquisition velocity, or data representation limits the ability to perform effective analysis using traditional relational approaches or requires the use of significant horizontal scaling for efficient processing.'' In particular, big data can be further categorized into big data science and big data frameworks. Big data science is ''the study of techniques covering the acquisition, conditioning, and evaluation of big data,'' whereas big data frameworks are ''software libraries along with their associated algorithms that enable distributed processing and analysis of big data problems across clusters of computer units''. An instantiation of one or more big data frameworks is known as big data infrastructure.

Concurrently, there has been much discussion in various industries and academia about what big data actually means [20], [21].

However, reaching a consensus about the definition of big data is difficult, if not impossible. A logical choice might be to embrace all the alternative definitions, each of which focuses on a specific aspect of big data. In this paper, we take this approach and embark on developing an understanding of common problems and approaches in big data science and engineering.

TABLE 1. Comparison between big data and traditional data.

	Traditional Data	Big Data
Volume	GB	constantly updated (TB or PB currently)
Generated Rate	per hour, day, ...	more rapid
Structure	structured	semi-structured or un-structured
Data Source	centralized	fully distributed
Data Integration	easy	difficult
Data Store	RDBMS	HDFS, NoSQL
Access	interactive	batch or near real-time

The aforementioned definitions for big data provide a set of tools to compare the emerging big data with traditional data analytics. This comparison is summarized in Table 1, under the framework of the ''4Vs''. First, the sheer volume of datasets is a critical factor for discriminating between big data and traditional data. For example, Facebook reports that its users registered 2.7 billion ''like'' and comments per day [22] in February 2012. Second, big data comes in three flavors: structured, semi-structured and unstructured. Traditional data are typically structured and can thus be easily tagged and stored. However, the vast majority of today's data, from sources such as Facebook, Twitter, YouTube and other user-generated content, are unstructured. Third, the velocity of big data means that datasets must be analyzed at a rate that matches the speed of data production. For time-sensitive applications, such as fraud detection and RFID data management, big data is injected into the enterprise in the form of a stream, which requires the system to process the data

stream as quickly as possible to maximize its value. Finally, by exploiting a variety of mining methods to analyze big datasets, significant value can be derived from a huge volume of data with a low value density in the form of deep insight or commercial benefits.

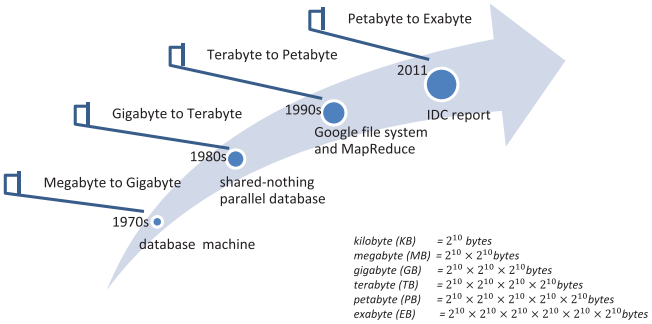


FIGURE 2. A brief history of big data with major milestones. It can be roughly split into four stages according to the data size growth of order, including Megabyte to Gigabyte, Gigabyte to Terabyte, Terabyte to Petabyte, and Petabyte to Exabyte.

B. A BRIEF HISTORY OF BIG DATA

Following its definition, we move to understanding the history of big data, i.e., how it evolved into its current stage. Considering the evolution and complexity of big data systems, previous descriptions are based on a one-sided viewpoint, such as chronology [23] or milepost technologies [24]. In this survey, the history of big data is presented in terms of the data size of interest. Under this framework, the history of big data is tied tightly to the capability of efficiently storing and managing larger and larger datasets, with size limitations expanding by orders of magnitude. Specifically, for each capability improvement, new database technologies were developed, as shown in Fig. 2. Thus, the history of big data can be roughly split into the following stages:

- *Megabyte to Gigabyte*: In the 1970s and 1980s, historical business data introduced the earliest "big data" challenge in moving from megabyte to gigabyte sizes. The urgent need at that time was to house that data and run relational queries for business analyses and reporting. Research efforts were made to give birth to the "database machine" that featured integrated hardware and software to solve problems. The underlying philosophy was that such integration would provide better performance at lower cost. After a period of time, it became clear that hardware-specialized database machines could not keep pace with the progress of general-purpose computers. Thus, the descendant database systems are software systems that impose few constraints on hardware and can run on general-purpose computers.
- *Gigabyte to Terabyte*: In the late 1980s, the popularization of digital technology caused data volumes to expand to several gigabytes or even a terabyte, which is beyond the storage and/or processing capabilities of a single large computer system. Data parallelization was proposed to extend storage capabilities and to improve performance by distributing data and related tasks, such as building indexes and evaluating queries, into disparate hardware. Based on this idea, several types of parallel databases were built, including shared-memory databases, shared-disk databases, and shared-nothing databases, all as induced by the underlying hardware architecture. Of the three types of databases, the shared-nothing architecture, built on a networked cluster of individual machines - each with its own processor, memory and disk [25] - has witnessed great success. Even in the past few years, we have witnessed the blooming of commercialized products of this type, such as Teradata [26], Netezza [27], Aster Data [28], Greenplum [29], and Vertica [30]. These systems exploit a relational data model and declarative relational query languages, and they pioneered the use of divide-and-conquer parallelism to partition data for storage.
- *Terabyte to Petabyte*: During the late 1990s, when the database community was admiring its "finished" work on the parallel database, the rapid development of Web 1.0 led the whole world into the Internet era, along with massive semi-structured or unstructured webpages holding terabytes or petabytes (PBs) of data. The resulting need for search companies was to index and query the mushrooming content of the web. Unfortunately, although parallel databases handle structured data well, they provide little support for unstructured data. Additionally, systems capabilities were limited to less than several terabytes. To address the challenge of web-scale data management and analysis, Google created Google File System (GFS) [31] and MapReduce [13] programming model. GFS and MapReduce enable automatic data parallelization and the distribution of large-scale computation applications to large clusters of commodity servers. A system running GFS and MapReduce can scale up and out and is therefore able to process unlimited data. In the mid-2000s, user-generated content, various sensors, and other ubiquitous data sources produced an overwhelming flow of mixed-structure data, which called for a paradigm shift in computing architecture and large-scale data processing mechanisms. NoSQL databases, which are scheme-free, fast, highly scalable, and reliable, began to emerge to handle these data. In Jan. 2007, Jim Gray, a database software pioneer, called the shift the "fourth paradigm" [32]. He also argued that the only way to cope with this paradigm was to develop a new generation of computing tools to manage, visualize and analyze the data deluge.
- *Petabyte to Exabyte*: Under current development trends, data stored and analyzed by big companies will undoubtedly reach the PB to exabyte magnitude soon. However, current technology still handles terabyte to PB data; there has been no revolutionary technology developed to cope with larger datasets. In Jun. 2011, EMC published a report entitled "Extracting Value from Chaos" [15].

The concept of big data and its potential were discussed throughout the report. This report ignited the enthusiasm for big data in industry and academia. In the years that followed, almost all the dominating industry companies, including EMC, Oracle, Microsoft, Google, Amazon, and Facebook, began to develop big data projects. In March 2012, the Obama administration announced that the US would invest 200 million dollars to launch a big data research plan. The effort will involve a number of federal agencies, including DARPA, the National Institutes of Health, and the National Science Foundation [8]. This endeavor aims to foster the development of advanced data management and analysis methods.

To look into future, we believe that data will continue to expand by orders of magnitude, and we are fortunate enough to stand in the initial stage of this big data wave [33], on which there are great opportunities to create revolutionary data management mechanisms or tools.

C. BIG-DATA PARADIGMS: STREAMING VS. BATCH

Big data analytics is the process of using analysis algorithms running on powerful supporting platforms to uncover potentials concealed in big data, such as hidden patterns or unknown correlations. According to the processing time requirement, big data analytics can be categorized into two alternative paradigms:

- *Streaming Processing*: The start point for the streaming processing paradigm [34] is the assumption that the potential value of data depends on data freshness. Thus, the streaming processing paradigm analyzes data as soon as possible to derive its results. In this paradigm, data arrives in a stream. In its continuous arrival, because the stream is fast and carries enormous volume, only a small portion of the stream is stored in limited memory. One or few passes over the stream are made to find approximation results. Streaming processing theory and technology have been studied for decades. Representative open source systems include Storm [35], S4 [36], and Kafka [37]. The streaming processing paradigm is used for online applications, commonly at the second, or even millisecond, level.
- *Batch Processing*: In the batch-processing paradigm, data are first stored and then analyzed. MapReduce [13] has become the dominant batch-processing model. The core idea of MapReduce is that data are first divided into small chunks. Next, these chunks are processed in parallel and in a distributed manner to generate intermediate results. The final result is derived by aggregating all the intermediate results. This model schedules computation resources close to data location, which avoids the communication overhead of data transmission. The MapReduce model is simple and widely applied in bioinformatics, web mining, and machine learning.

There are many differences between these two processing paradigms, as summarized in Table 2. In general, the streaming processing paradigm is suitable for applications in which data are generated in the form of a stream and rapid processing is required to obtain approximation results. Therefore, the streaming processing paradigm's application domains are relatively narrow. Recently, most applications have adopted the batch-processing paradigm; even certain real-time processing applications use the batch-processing paradigm to achieve a faster response. Moreover, some research effort has been made to integrate the advantages of these two paradigms.

TABLE 2. Comparison between streaming processing and batch processing.

	streaming processing	batch processing
Input	stream of new data or updates	data chunks
Data size	infinite or unknown in advance	known & finite
Storage	not store or store non-trival portion in memory	store
Hardware	typical single limited amount of memory	multiple CPUs, memories
Processing	a single or few pass(es) over data	processed in multiple rounds
Time	a few seconds or even milliseconds	much longer
Applications	web mining, sensor networks, traffic monitoring	widely adopted in almost every domain

Big data platforms can use alternative processing paradigms; however, the differences in these two paradigms will cause architectural distinctions in the associated platforms. For example, batch-processing-based platforms typically encompass complex data storage and management systems, whereas streaming-processing-based platforms do not. In practice, we can customize the platform according to the data characteristics and application requirements. Because the batch-processing paradigm is widely adopted, we only consider batch-processing- based big data platforms in this paper.

III. BIG-DATA SYSTEM ARCHITECTURE

In this section, we focus on the value chain for big data analytics. Specifically, we describe a big data value chain that consists of four stages (generation, acquisition, storage, and processing). Next, we present a big data technology map that associates the leading technologies in this domain with specific phases in the big data value chain and a time stamp.

A. BIG-DATA SYSTEM: A VALUE-CHAIN VIEW

A big-data system is complex, providing functions to deal with different phases in the digital data life cycle, ranging from its birth to its destruction. At the same time, the system usually involves multiple distinct phases for different applications [38], [39]. In this case, we adopt a systems-engineering approach, well accepted in industry, [40], [41] to decompose a typical big-data system into four consecutive phases, including data generation, data acquisition, data storage, and data analytics, as illustrated in the horizontal axis of Fig. 3. Notice that data visualization is an assistance method for data analysis. In general, one shall visualize data to find some

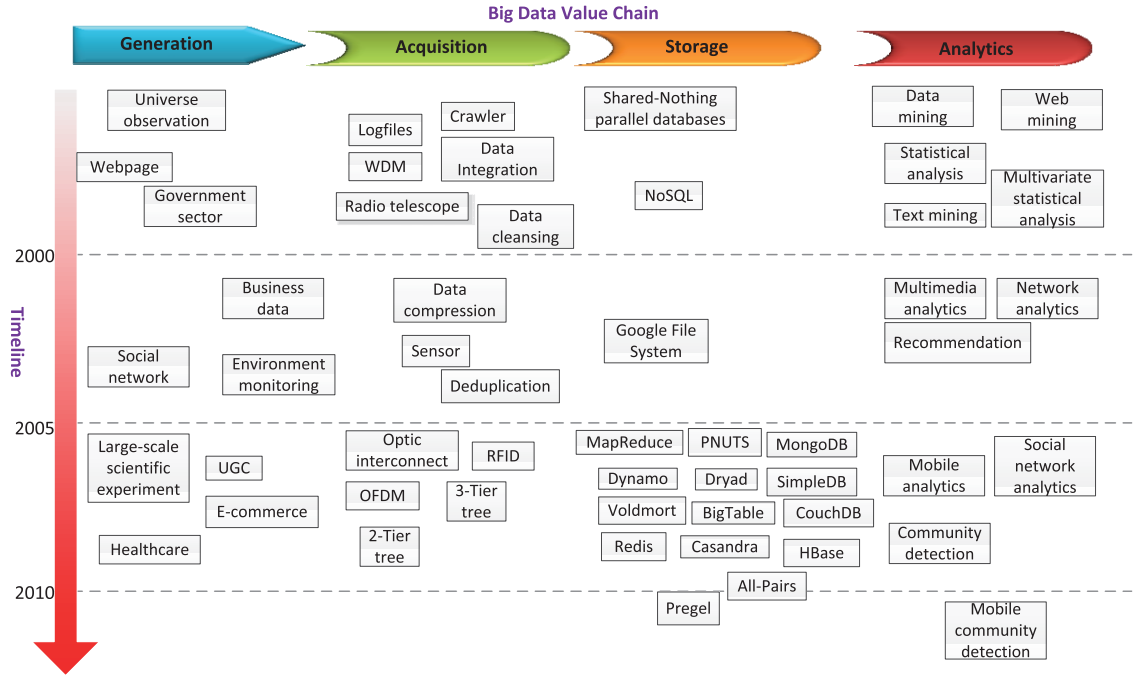


FIGURE 3. **Big data technology map. It pivots on two axes, i.e., data value chain and timeline. The data value chain divides the data lifecycle into four stages, including data generation, data acquisition, data storage, and data analytics. In each stage, we highlight exemplary technologies over the past 10 years.**

rough patterns first, and then employ specific data mining methods. I mention this in data analytics section. The details for each phase are explained as follows.

Data generation concerns how data are generated. In this case, the term "big data" is designated to mean large, diverse, and complex datasets that are generated from various longitudinal and/or distributed data sources, including sensors, video, click streams, and other available digital sources. Normally, these datasets are associated with different levels of domain-specific values [2]. In this paper, we focus on datasets from three prominent domains, business, Internet, and scientific research, for which values are relatively easy to understand. However, there are overwhelming technical challenges in collecting, processing, and analyzing these datasets that demand new solutions to embrace the latest advances in the information and communications technology (ICT) domain.

Data acquisition refers to the process of obtaining information and is subdivided into data collection, data transmission, and data pre-processing. First, because data may come from a diverse set of sources, websites that host formatted text, images and/or videos - data collection refers to dedicated data collection technology that acquires raw data from a specific data production environment. Second, after collecting raw data, we need a high-speed transmission mechanism to transmit the data into the proper storage sustaining system for various types of analytical applications. Finally, collected datasets might contain many meaningless data, which unnecessarily increases the amount of storage space and affects the consequent data analysis. For instance, redundancy is common in most datasets collected from sensors deployed to monitor the environment, and we can use data compression technology to address this issue. Thus, we must perform data pre-processing operations for efficient storage and mining.

Data storage concerns persistently storing and managing large-scale datasets. A data storage system can be divided into two parts: hardware infrastructure and data management. Hardware infrastructure consists of a pool of shared ICT resources organized in an elastic way for various tasks in response to their instantaneous demand. The hardware infrastructure should be able to scale up and out and be able to be dynamically reconfigured to address different types of application environments. Data management software is deployed on top of the hardware infrastructure to maintain large-scale datasets. Additionally, to analyze or interact with the stored data, storage systems must provide several interface functions, fast querying and other programming models.

Data analysis leverages analytical methods or tools to inspect, transform, and model data to extract value. Many application fields leverage opportunities presented by abundant data and domain-specific analytical methods to derive the intended impact. Although various fields pose different application requirements and data characteristics, a few of these fields may leverage similar underlying technologies. Emerging analytics research can be classified into six critical technical areas: structured data analytics, text analytics, multimedia analytics, web analytics, network analytics, and mobile analytics. This classification is

intended to highlight the key data characteristics of each area.

B. BIG-DATA TECHNOLOGY MAP

Big data research is a vast field that connects with many enabling technologies. In this section, we present a big data technology map, as illustrated in Fig. 3. In this technology map, we associate a list of enabling technologies, both open-source and proprietary, with different stages in the big data value chain.

This map reflects the development trends of big data. In the data generation stage, the structure of big data becomes increasingly complex, from structured or unstructured to a mixture of different types, whereas data sources become increasingly diverse. In the data acquisition stage, data collection, data pre-processing, and data transmission research emerge at different times. Most research in the data storage stage began in approximately 2005. The fundamental methods of data analytics were built before 2000, and subsequent research attempts to leverage these methods to solve domain-specific problems. Moreover, qualified technology or methods associated with different stages can be chosen from this map to customize a big data system.

C. BIG-DATA SYSTEM: A LAYERED VIEW

Alternatively, the big data system can be decomposed into a layered structure, as illustrated in Fig. 4. The layered structure is divisible into three layers, i.e., the infrastructure layer, the computing layer, and the application layer, from bottom to top. This layered view only provides a conceptual hierarchy to underscore the complexity of a big data system. The function of each layer is as follows.

- The *infrastructure layer* consists of a pool of ICT resources, which can be organized by cloud computing infrastructure and enabled by virtualization technology. These resources will be exposed to upper-layer systems in a fine-grained manner with a specific service-level agreement (SLA). Within this model, resources must be allocated to meet the big data demand while achieving resource efficiency by maximizing system utilization, energy awareness, operational simplification, etc.
- The *computing layer* encapsulates various data tools into a middleware layer that runs over raw ICT resources. In the context of big data, typical tools include data integration, data management, and the programming model. Data integration means acquiring data from disparate sources and integrating the dataset into a unified form with the necessary data pre-processing operations. Data management refers to mechanisms and tools that provide persistent data storage and highly efficient management, such as distributed file systems and SQL or NoSQL data stores. The programming model implements abstraction application logic and facilitates the data analysis applications. MapReduce [13], Dryad [42], Pregel [43], and Dremel [44] exemplify programming models.
- The *application layer* exploits the interface provided by the programming models to implement various data analysis functions, including querying, statistical analyses, clustering, and classification; then, it combines basic analytical methods to develop various filed related applications. McKinsey presented five potential big data application domains: health care, public sector administration, retail, global manufacturing, and personal location data.

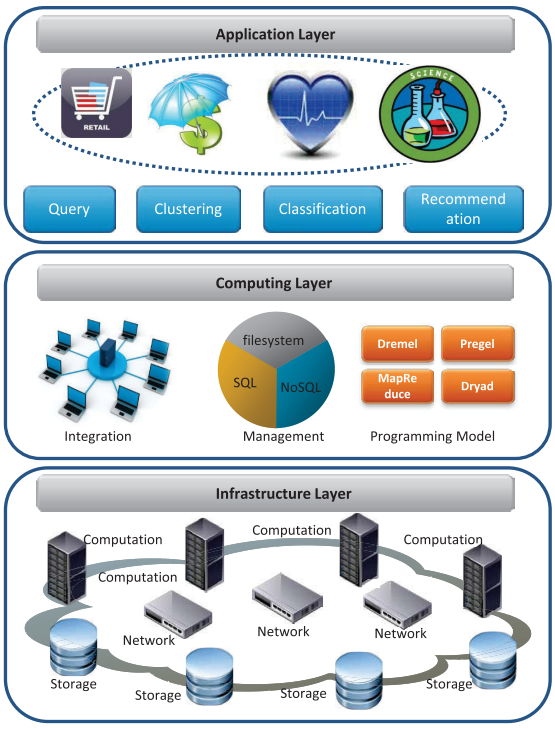


FIGURE 4. Layered architecture of big data system. It can be decomposed into three layers, including infrastructure layer, computing layer, and application layer, from bottom to up.

D. BIG-DATA SYSTEM CHALLENGES

Designing and deploying a big data analytics system is not a trivial or straightforward task. As one of its definitions suggests, big data is beyond the capability of current hardware and software platforms. The new hardware and software platforms in turn demand new infrastructure and models to address the wide range of challenges of big data. Recent works [38], [45], [46] have discussed potential obstacles to the growth of big data applications. In this paper, we strive to classify these challenges into three categories: data collection and management, data analytics, and system issues.

Data collection and management addresses massive amounts of heterogeneous and complex data. The following challenges of big data must be met:

- *Data Representation*: Many datasets are heterogeneous in type, structure, semantics, organization, granularity, and accessibility. A competent data presentation should be designed to reflect the structure, hierarchy,

and diversity of the data, and an integration technique should be designed to enable efficient operations across different datasets.

- *Redundancy Reduction and Data Compression*: Typically, there is a large number of redundant data in raw datasets. Redundancy reduction and data compression without scarifying potential value are efficient ways to lessen overall system overhead.
- *Data Life-Cycle Management*: Pervasive sensing and computing is generating data at an unprecedented rate and scale that exceed much smaller advances in storage system technologies. One of the urgent challenges is that the current storage system cannot host the massive data. In general, the value concealed in the big data depends on data freshness; therefore, we should set up the data importance principle associated with the analysis value to decide what parts of the data should be archived and what parts should be discarded.
- *Data Privacy and Security*: With the proliferation of online services and mobile phones, privacy and security concerns regarding accessing and analyzing personal information is growing. It is critical to understand what support for privacy must be provided at the platform level to eliminate privacy leakage and to facilitate various analyses.

There will be a significant impact that results from advances in big data analytics, including interpretation, modeling, prediction, and simulation. Unfortunately, massive amounts of data, heterogeneous data structures, and diverse applications present tremendous challenges, such as the following.

- *Approximate Analytics*: As data sets grow and the real-time requirement becomes stricter, analysis of the entire dataset is becoming more difficult. One way to potentially solve this problem is to provide approximate results, such as by means of an approximation query. The notion of approximation has two dimensions: the accuracy of the result and the groups omitted from the output.
- *Connecting Social Media*: Social media possesses unique properties, such as vastness, statistical redundancy and the availability of user feedback. Various extraction techniques have been successfully used to identify references from social media to specific product names, locations, or people on websites. By connecting inter-field data with social media, applications can achieve high levels of precision and distinct points of view.
- *Deep Analytics*: One of the drivers of excitement around big data is the expectation of gaining novel insights. Sophisticated analytical technologies, such as machine learning, are necessary to unlock such insights. However, effectively leveraging these analysis toolkits requires an understanding of probability and statistics. The potential pillars of privacy and security mechanisms are mandatory access control and security communication, multi-granularity access control, privacy-aware data mining and analysis, and security storage and management.

Finally, large-scale parallel systems generally confront several common issues; however, the emergence of big data has amplified the following challenges, in particular.

- *Energy Management*: The energy consumption of large-scale computing systems has attracted greater concern from economic and environmental perspectives. Data transmission, storage, and processing will inevitably consume progressively more energy, as data volume and analytics demand increases. Therefore, system-level power control and management mechanisms must be considered in a big data system, while continuing to provide extensibility and accessibility.
- *Scalability*: A big data analytics system must be able to support very large datasets created now and in the future. All the components in big data systems must be capable of scaling to address the ever-growing size of complex datasets.
- *Collaboration*: Big data analytics is an interdisciplinary research field that requires specialists from multiple professional fields collaborating to mine hidden values. A comprehensive big data cyber infrastructure is necessary to allow broad communities of scientists and engineers to access the diverse data, apply their respective expertise, and cooperate to accomplish the goals of analysis.

In the remainder of this paper, we follow the value-chain framework illustrated in Fig. 3 to investigate the four phases of the big-data analytic platform.

IV. PHASE I: DATA GENERATION

In this section, we present an overview of two aspects of big data sources. First, we discuss the historical trends of big data sources and then focus on three typical sources of big data. Following this, we use five data attributes introduced by the National Institute of Standards and Technology (NIST) to classify big data.

A. DATA SOURCES: TRENDS AND EXEMPLARY CATEGORIES

The trends of big data generation can be characterized by the data generation rate. Specifically, the data generation rate is increasing due to technological advancements. Indeed, IBM estimated that 90% of the data in the world today has been created in the past two years [47]. The cause of the data explosion has been much debated. Cisco argued that the growth is caused mainly by video, the Internet, and cameras [48]. Actually, data refers to the abstraction of information that is readable by a computer. In this sense, ICT is the principal driving force that makes information readable and creates or captures data. In this paper, therefore, we begin our discussion with the development of ICT and take a historical perspective in explaining the data explosion trend. Specifically, we roughly classify data generation patterns into three

sequential stages:

- *Stage I*: The first stage began in the 1990s. As digital technology and database systems were widely adopted, many management systems in various organizations were storing large volumes of data, such as bank trading transactions, shopping mall records, and government sector archives. These datasets are structured and can be analyzed through database-based storage management systems.
- *Stage II*: The second stage began with the growing popularity of web systems. The Web 1.0 systems, characterized by web search engines and ecommerce businesses after the late 1990s, generated large amounts of semi-structured and/or unstructured data, including webpages and transaction logs. Since the early 2000s, many Web 2.0 applications created an abundance of user-generated content from online social networks, such as forums, online groups, blogs, social networking sites, and social media sites.
- *Stage III*: The third stage is triggered by the emergence of mobile devices, such as smart phones, tablets, sensors and sensor-based Internet-enabled devices. The mobile-centric network has and will continue to create highly mobile, location-aware, person-centered, and context-relevant data in the near future.

With this classification, we can see that the data generation pattern is evolving rapidly, from passive recording in Stage I to active generation in Stage II and automatic production in Stage III. These three types of data constitute the primary sources of big data, of which the automatic production pattern will contribute the most in the near future.

In addition to its generic property (e.g., its rate of generation), big data sources are tightly coupled with their generating domains. In fact, exploring datasets from different domains may create distinctive levels of potential value [2]. However, the potential domains are so broad that they deserve their own dedicated survey paper. In this survey, we mainly focus on datasets from the following three domains to investigate big data-related technologies: business, networking, and scientific research. Our reasons of choice are as follows. First, big data is closely related to business operations and many big data tools have thus previously been developed and applied in industry. Second, most data remain closely bound to the Internet, the mobile network and the Internet of Things. Third, as scientific research generates more data, effective data analysis will help scientists reveal fundamental principles and hence boost scientific development. The three domains vary in their sophistication and maturity in utilizing big data and therefore might dictate different technological requirements.

1) BUSINESS DATA

The use of information technology and digital data has been instrumental in boosting the profitability of the business sector for decades. The volume of business data worldwide across all companies is estimated to double every 1.2 years [49]. Business transactions on the Internet, including business-to-business and business-to-consumer transactions, will reach 450 billion per day [50]. The ever-increasing volume of business data calls for more effective real-time analysis to gain further benefits. For example, every day, Amazon handles millions of back-end operations and queries from more than half a million third-party sellers [51]. Walmart handles more than 1 million customer transactions every hour. These transactions are imported into databases that are estimated to contain more than 2.5 PBs of data [3]. Akamai analyzes 75 million events per day to better target advertisements [9].

2) NETWORKING DATA

Networking, including the Internet, the mobile network, and the Internet of Things, has penetrated into human lives in every possible aspect. Typical network applications, regarded as the network big data sources, include, but are not limit to, search, SNS, websites, and click streams. These sources are generating data at record speeds, demanding advanced technologies. For example, Google, a representative search engine, was processing 20 PBs a day in 2008 [13]. For social network applications, Facebook stored, accessed, and analyzed more than 30 PBs of user-generated data. Over 32 billion searches were performed per month on Twitter [52]. In the mobile network field, more than 4 billion people, or 60 percent of the world's population, were using mobile phones in 2010, and approximately 12 percent of these people had smart phones [2]. In the field of the Internet of Things, more than 30 million networked sensor nodes are now functioning in the transportation, automotive, industrial, utilities, and retail sectors. The number of these sensors is increasing at a rate of more than 30 percent per year [2].

3) SCIENTIFIC DATA

More and more scientific applications are generating very large datasets, and the development of several disciplines greatly relies on the analysis of massive data. In this domain, we highlight three scientific domains that are increasingly relying on big data analytics:

- *Computational Biology:* The National Center for Biotechnology Innovation maintains the GenBank database of nucleotide sequences, which doubles in size every 10 months. As of August 2009, the database contains over 250 billion nucleotide bases from more than 150,000 distinct organisms [53].
- *Astronomy:* From 1998 to 2008, the Sloane Digital Sky Survey (SDSS), the largest astronomical catalogue, generated 25 terabytes of data from telescopes. As tele-

TABLE 3. Typical big data sources.

Data Source	Application	Data Scale	Type	Response Time	Number of Users	Accurancy
Walmart	retail	PB	structured	very fast	large	very high
Amazon	e-commerce	PB	semi-structured	very fast	large	very high
Google search	Internet	PB	semi-structured	fast	very large	high
facebook	Social network	PB	structured, un-structured	fast	very large	high
AT&T	Mobile network	TB	structured	fast	very large	high
healthe care	Internet of Things	TB	structured, un-structured	fast	large	high
SDSS	Scientific	TB	un-structured	slow	small	very high

scope resolutions have increased, the generated data volume each night is anticipated to exceed 20 terabytes in 2014 [54].

- *High-Energy Physics:* The Atlas experiment for the Large Hadron Collider at the Center for European Nuclear Research will generate raw data at a rate of 2 PBs per second at the beginning of 2008 and will store approximately 10 PBs per year of processed data [55].

These areas not only generate a huge amount of data but also require multiple geo-distributed parties to collaborate on analyzing the data [56], [57]. Interested scholars should refer to the important discussions on data science [32], including earth and environment, health and well-being, scientific infrastructure, and scholarly communication.

In Table 3, we enumerate representative big data sources from these three domains and their attributes from the application and analysis requirement perspective. As can be easily shown, most data sources generate PB level unstructured data, which requires fast and accurate analysis for a large number of users.

B. DATA ATTRIBUTES

Pervasive sensing and computing across natural, business, Internet and government sectors and social environments are generating heterogeneous data with unprecedented complexity. These datasets may have distinctive data characteristics in terms of scale, temporal dimensional, or variety of data types. For example, in [58], mobile data types related to location, motion, proximity, communication, multimedia, application usage and audio environment were recorded. NIST [19] introduces five attributes to classify big data, which are listed below.

- *Volume* is the sheer volume of datasets.
- *Velocity* the data generation rate and real-time requirement.
- *Variety* refers to the data form, i.e., structured, semi-structured, and unstructured.
- *Horizontal Scalability* is the ability to join multiple datasets.
- *Relational Limitation* includes two categories, special forms of data and particular queries. Special forms of data include temporal data and spatial data. Particular queries may be recursive or another type.

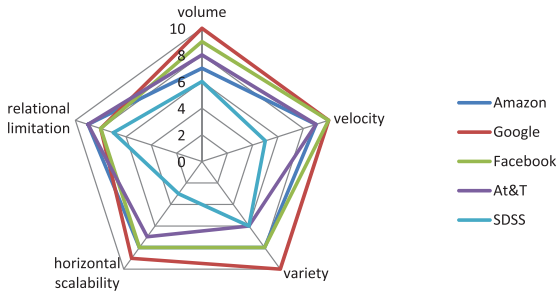


FIGURE 5. Five metrics to classify big data. These metrics are introduced by NIST [19], including volume, velocity, variety, horizontal scalability, and relational limitation.

Within this measure, we introduce a visualization tool, which is shown in Fig. 5. We can see that the data source from the scientific domain has the lowest attribute values in all aspects; data sources from the business domain have a higher horizontal scalability and relational limitation requirements, whereas data source from the networking domain have higher volume, velocity, and variety characteristics.

V. PHASE II: DATA ACQUISITION

As illustrated in the big data value chain, the task of the data acquisition phase is to aggregate information in a digital form for further storage and analysis. Intuitively, the acquisition process consists of three sub-steps, data collection, data transmission, and data pre-processing, as illustrated in Fig. 6. There is no strict order between data transmission and data pre-processing; thus, data pre-processing operations can occur before data transmission and/or after data transmission. In this section we review ongoing scholarship and current solutions for these three sub-tasks.

A. DATA COLLECTION

Data collection refers to the process of retrieving raw data from real-world objects. The process needs to be well designed. Otherwise, inaccurate data collection would impact the subsequent data analysis procedure and ultimately lead to invalid results. At the same time, data collection methods not only depend on the physics characteristics of data sources, but also the objectives of data analysis. As a result, there are many kinds of data collection methods. In the subsection, we will first focus on three common methods for big data collection, and then touch upon a few other related methods.

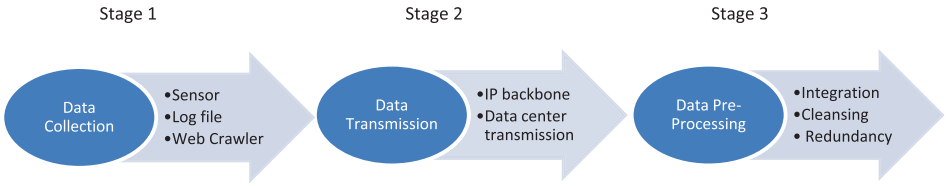


FIGURE 6. The Data acquisition stage consists of three sub-tasks: collection, transmission and pre-processing. In each stage, representative methods will be investigated. For example, the data collection stage covers three common methods, including sensor, log file, and web crawler.

1) SENSOR

Sensors are used commonly to measure a physical quantity and convert it into a readable digital signal for processing (and possibly storing). Sensor types include acoustic, sound, vibration, automotive, chemical, electric current, weather, pressure, thermal, and proximity. Through wired or wireless networks, this information can be transferred to a data collection point.

Wired sensor networks leverage wired networks to connect a collection of sensors and transmit the collected information. This scenario is suitable for applications in which sensors can easily be deployed and managed. For example, many video surveillance systems in industry are currently built using a single Ethernet unshielded twisted pair per digital camera wired to a central location (certain systems may provide both wired and wireless interfaces) [59]. These systems can be deployed in public spaces to monitor human behavior, such as theft and other criminal behaviors.

By contrast, wireless sensor networks (WSNs) utilize a wireless network as the substrate of information transmission. This solution is preferable when the exact location of a particular phenomenon is unknown, particularly when the environment to be monitored does not have an infrastructure for either energy or communication. Recently, WSNs have been widely discussed and applied in many applications, such as in environment research [60], [61], water monitoring [62], civil engineering [63], [64], and wildlife habitat monitoring [65]. The WSN typically consists of a large number of spatially distributed sensor nodes, which are battery-powered tiny devices. Sensors are first deployed at the locations specified by the application requirement to collect sensing data. After sensor deployment is complete, the base station will disseminate the network setup/management and/or collection command messages to all sensor nodes. Based on this indicated information, sensed data are gathered at different sensor nodes and forwarded to the base station for further processing. [66] offer a detailed discussion of the foregoing.

A sensor based data collection system can be considered as a cyber-physical system [67]. Actually, in the scientific experiment domain, many specialty instruments, such as magnetic spectrometer, radio telescope, are used to collect experiment data [68]. They can be regarded as a special type of sensor. In this sense, experiment data collection systems also belong to the category of cyber-physical system.

A sensor-based data collection system is considered a cyber-physical system [67]. In the scientific experiment domain, many specialty instruments, such as magnetic spectrometers and radio telescopes, are used to collect experimental data [68]. These instruments may be considered as a special type of sensor. In this sense, experiment data collection systems also belong to the category of cyber-physical systems.

2) LOG FILE

Log files, one of the most widely deployed data collection methods, are generated by data source systems to record activities in a specified file format for subsequent analysis. Log files are useful in almost all the applications running on digital devices. For example, a web server normally records all the clicks, hits, access and other attributes [69] made by any website user in an access log file. There are three main types of web server log file formats available to capture the activities of users on a website: Common Log File Format (NCSA), Extended Log Format (W3C), and IIS Log Format (Microsoft). All three log file formats are in the ASCII text format. Alternatively, databases can be utilized instead of text files to store log information to improve the querying efficiency of massive log repositories [70], [71]. Other examples of log file-based data collection include stock ticks in financial applications, performance measurement in network monitoring, and traffic management.

In contrast to a physical sensor, a log file can be viewed as ''software-as-a-sensor''. Much user-implemented data collection software [58] belongs to this category.

3) WEB CRAWLER

A crawler [72] is a program that downloads and stores webpages for a search engine. Roughly, a crawler starts with an initial set of URLs to visit in a queue. All the URLs to be retrieved are kept and prioritized. From this queue, the crawler gets a URL that has a certain priority, downloads the page, identifies all the URLs in the downloaded page, and adds the new URLs to the queue. This process is repeated until the crawler decides to stop. Web crawlers are general data collection applications for website-based applications, such as web search engines and web caches. The crawling process is determined by several policies, including the selection policy, re-visit policy, politeness policy, and parallelization policy [73]. The selection policy communicates which pages to download; the re-visit policy decides when to check for changes to the pages; the politeness policy prevents overloading the websites; the parallelization policy coordinates distributed web crawlers. Traditional web application crawling

TABLE 4. Comparison for three data collection methods.

method	mode	data structure	data scale	complexity	typical applications
sensor	pull	structured or unstrucutred	median	sophisticated	video surveillance, inventory management
log file	push	structured or semi-structured	small	easy	web log, click stream
web crawler	pull	mixture	large	median	search, SNS analysis

is a well-researched field with multiple efficient solutions. With the emergence of richer and more advanced web applications, some crawling strategies [74] have been proposed to crawl rich Internet applications. Currently, there are plenty of general-purpose crawlers available as enumerated in the list [75].

In addition to the methods discussed above, there are many data collection methods or systems that pertain to specific domain applications. For example, in certain government sectors, human biometrics [76], such as fingerprints and signatures, are captured and stored for identity authentication and to track criminals. In summary, data collection methods can be roughly divided into two categories:

- *Pull-Based Approach*: Data are collected proactively by a centralized/distributed agent.
- *Push-Based Approach*: Data are pushed toward the sink by its source or a third party.

The three aforementioned methods are compared in Table 4. We can see from the table that the log file is the simplest data collection method, but it can collect only a relatively small amount of structured data. The web crawler is the most flexible data collection model and can acquire vast amounts of data with complex structures.

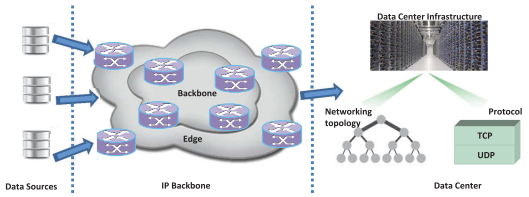


FIGURE 7. Big data transmission procedure. It can be divided into two stages, IP backbone transmission and data center transmission.

B. DATA TRANSMISSION

Once we gather the raw data, we must transfer it into a data storage infrastructure, commonly in a data center, for subsequent processing. The transmission procedure can be divided into two stages, IP backbone transmission and data center transmission, as illustrated in Fig. 7. Next, we introduce several emerging technologies in these two stages.

1) IP BACKBONE

The IP backbone, at either the region or Internet scale, provides a high-capacity trunk line to transfer big data from its origin to a data center. The transmission rate and capacity are determined by the physical media and the link management methods.

- *Physical Media* are typically composed of many fiber optic cables bundled together to increase capacity. In general, physical media should guarantee path diversity to reroute traffic in case of failure.
- *Link Management* concerns how the signal is transmitted over the physical media. IP over Wavelength-Division Multiplexing (WDM) has been developed over the past two decades [77], [78]. WDM is technology that multiplexes multiple optical carrier signals on a single optical fiber using different wavelengths of laser light to carry different signals. To address the electrical bandwidth bottleneck limitation, Orthogonal Frequency-Division Multiplexing (OFDM) has been considered as a promising candidate for future high-speed optical transmission technology. OFDM allows the spectrum of individual subcarriers to overlap, which leads to a more data-rate flexible, agile, and resource-efficient optical network [79], [80].

Thus far, optical transmission systems with up to capacities of 40 Gb/s per channel have been deployed in backbone networks, whereas 100 Gb/s interfaces are now commercially available and 100 Gb/s deployment is expected soon. Even Tb/s-level transmission is foreseen in the near future [81].

Due to the difficulty of deploying enhanced network protocols in the Internet backbone, we must follow standard Internet protocols to transmit big data. However, for a regional or private IP backbone, certain alternatives [82] may achieve better performance for specific applications.

2) DATA CENTER TRANSMISSION

When big data is transmitted into the data center, it will be transited within the data center for placement adjustment, processing, and so on. This process is referred to as data center transmission. It always associates with data center network architecture and transportation protocol:

- *Data Center Network Architecture*: A data center consists of multiple racks hosting a collection of servers connected through the data center's internal connection network. Most current data center internal connection networks are based on commodity switches that configure a canonical fat-tree 2-tier [83] or 3-tier architecture [84]. Some other topologies that aim to create more efficient data center networks can be found in [85]–[88]. Because of the inherent shortage of electronic packet switches, increasing communication bandwidth while simultaneously reducing energy consumption is difficult. Optical interconnects for data center networks have gained attention recently as a promising solution that offers high throughput, low latency, and reduced

energy consumption. Currently, optical technology has been adopted in data centers only for point-to-point links. These links are based on low-cost multi-mode fibers (MMF) for the connections of switches, with bandwidths up to 10 Gbps [89]. The use of optical interconnects for data center networks [90] (in which the switching is performed at the optical domain) is a viable solution for providing Tbps transmission bandwidths with increased energy efficiency. Many optical interconnect schemes [87], [91]–[95] have recently been proposed for data center networks. Certain schemes add optical circuits to upgrade current networks, whereas other schemes completely replace the current switches. However, more effort is required to make these novel technologies mature.

- *Transportation Protocol*: TCP and UDP are the most important network protocols for data transmission; however, their performance is not satisfactory when there is a large amount of data to be transferred. Much research effort was made to improve the performance of these two protocols. Enhanced TCP methods aim to improve link throughput while providing a small predictable latency for a diverse mix of short and long TCP flows. For instance, DCTCP [96] leverages Explicit Congestion Notification (ECN) in the network to provide multi-bit feedback to the end host, allowing it the host to react early to congestion. Vamanan et al. [97] proposed a deadline-aware data center TCP for bandwidth allocation, which can guarantee that network communication is finished under soft real-time constraints. UDP is suitable for transferring a huge volume of data but lacks congestion control, unfortunately. Thus, high bandwidth UDP applications must implement congestion control themselves, which is a difficult task and may incur risk, which renders congested networks unusable. Kohler et al. [98] designed a congestion-controlled unreliable transport protocol, adding to a UDP-like foundation to support congestion control. This protocol resembles TCP but without reliability and cumulative acknowledgements.

C. DATA PRE-PROCESSING

Because of their diverse sources, the collected data sets may have different levels of quality in terms of noise, redundancy, consistency, etc. Transferring and storing raw data would have necessary costs. On the demand side, certain data analysis methods and applications might have strict requirements on data quality. As such, data preprocessing techniques that are designed to improve data quality should be in place in big data systems. In this subsection, we briefly survey current research efforts for three typical data pre-processing techniques. [99]–[101] provide a more in-depth treatment of this topic.

1) INTEGRATION

Data integration techniques aim to combine data residing in different sources and provide users with a unified view of the data [102]. Data integration is a mature field in traditional database research [103]. Previously, two approaches prevailed, the data warehouse method and the data federation method. The data warehouse method [102], also known as ETL, consists of the following three steps: extraction, transformation and loading.

- The extraction step involves connecting to the source systems and selecting and collecting the necessary data for analysis processing.
- The transformation step involves the application of a series of rules to the extracted data to convert it into a standard format.
- The load step involves importing extracted and transformed data into a target storage infrastructure.

Second, the data federation method creates a virtual database to query and aggregate data from disparate sources. The virtual database does not contain data itself but instead contains information or metadata about the actual data and its location.

However, the ''store-and-pull'' nature of these two approaches is unsuitable for the high performance needs of streaming or search applications, where data are much more dynamic than the queries and must be processed on the fly. In general, data integration methods are better intertwined with the streaming processing engines [34] and search engines [104].

2) CLEANSING

The data cleansing technique refers to the process to determine inaccurate, incomplete, or unreasonable data and then to amend or remove these data to improve data quality. A general framework [105] for data cleansing consists of five complementary steps:

- Define and determine error types;
- Search and identify error instances;
- Correct the errors;
- Document error instances and error types; and
- Modify data entry procedures to reduce future errors.

Moreover, format checks, completeness checks, reasonableness checks, and limit checks [105] are normally considered in the cleansing process. Data cleansing is generally considered to be vital to keeping data consistent and updated [101] and is thus widely used in many fields, such as banking, insurance, retailing, telecommunications, and transportation.

Current data-cleaning techniques are spread across different domains. In the e-commerce domain, although most of the data are collected electronically, there can be serious data quality issues. Typical sources of such data quality issues include software bugs, customization mistakes, and the system configuration process. Kohavi et al. in [106] discusses cleansing e-commerce data by detecting crawlers and performing regular de-duping of customers and accounts.

In the radio frequency identification (RFID) domain, the work in [107] considers data cleansing for RFID data. RFID technologies are used in many applications, such as inventory checking and object tracking. However, raw RFID data are typically of low quality and may contain many anomalies because of physical device limitations and different types of environmental noise. In [108], Zhao et al. developed a probabilistic model to address the missing data problem in a mobile environment. Khoussainova et al. [109] presented a system for correcting input data errors automatically with application-defined global integrity constraints.

Another example was reported in [110] that implemented a framework, BIO-AJAX, to standardize biological data for further computation and to improve searching quality. With the help of BIO-AJ Ax, errors and duplicate can be eliminated, and common data-mining techniques will run more effectively.

Data cleansing is necessary for subsequent analysis because it improves analysis accuracy. However, data cleansing commonly depends on the complex relationship model and it incurs extra computation and delay overhead. We must seek a balance between the complexity of the data-cleansing model and the resulting improvement in the accuracy analysis.

3) REDUNDANCY ELIMINATION

Data redundancy is the repetition or superfluity of data, which is a common issue for various datasets. Data redundancy unnecessarily increases data transmission overhead and causes disadvantages for storage systems, including wasted storage space, data inconsistency, reduced reliability and data corruption. Therefore, many researchers have proposed various redundancy reduction methods, such as redundancy detection [111] and data compression [112]. These methods can be used for different datasets or application conditions and can create significant benefits, in addition to risking exposure to several negative factors. For instance, the data compression method poses an extra computational burden in the data compression and decompression processes. We should assess the tradeoff between the benefits of redundancy reduction and the accompanying burdens.

Data redundancy is exemplified by the growing amount of image and video data, collected from widely deployed cameras [113]. In the video surveillance domain, vast quantities of redundant information, including temporal redundancy, spatial redundancy, statistical redundancy and perceptual redundancy, is concealed in the raw image and video files [113]. Video compression techniques are widely used to reduce redundancy in video data. Many important standards (e.g., MPEG-2, MPEG-4, H.263, H.264/AVC) [114] have been built and applied to alleviate the burden on transmission and storage. In [115], Tsai et al. studied video compression for intelligent video surveillance via video sensor networks. By exploring the contextual redundancy associated with background and foreground objects in a scene, a novel approach beyond MPEG-4 and traditional methods has been proposed. In addition, further evaluation results reveal the low complexity and compression ratio of the approach.

For generalized data transmission or storage, the data deduplication [116] technique is a specialized data compression technique for eliminating duplicate copies of repeating data. In a storage deduplication process, a unique chunk or segment of data will be allocated an identification (e.g., hashing) and stored, and the identification will be added to an identification list. As the deduplication analysis continues, a new chunk associate with the identification, which already exists in the identification list, is regarded as a redundant chunk. This chunk is replaced with a reference that points to the stored chunk. In this way, only one instance of any piece of given data is retained. Deduplication can greatly reduce the amount of storage space and is particularly important for big data storage systems.

In addition to the data pre-processing methods described above, other operations are necessary for specific data objects. One example is feature extraction, which plays a critical role in areas such as multimedia search [117] and DNA analysis [118], [119]. Typically, these data objects are described by high-dimensional feature vectors (or points), which are organized in storage systems for retrieval. Another example is data transformation [120], which is typically used to handle distributed data sources with heterogeneous schema and is particularly useful for business datasets. Gunter et al. [121] developed a novel approach, called MapLan, to map and transform survey information from the Swiss National Bank.

However, no unified data pre-processing procedure and no single technique can be expected to work best across a wide variety of datasets. We must consider together the characteristics of the datasets, the problem to be solved, performance requirements and other factors to choose a suitable data pre-processing scheme.

VI. PHASE III: DATA STORAGE

The data storage subsystem in a big data platform organizes the collected information in a convenient format for analysis and value extraction. For this purpose, the data storage subsystem should provide two sets of features:

- The storage infrastructure must accommodate information persistently and reliably.
- The data storage subsystem must provide a scalable access interface to query and analyze a vast quantity of data.

This functional decomposition shows that the data storage subsystem can be divided into hardware infrastructure and data management. These two components are explained below.

A. STORAGE INFRASTRUCTURE

Hardware infrastructure is responsible for physically storing the collected information. The storage infrastructure can be understood from different perspectives.

First, storage devices can be classified based on the specific

technology. Typical storage technologies include, but are not limited to, the following.

- *Random Access Memory (RAM):* RAM is a form of computer data storage associated with volatile types of memory, which loses its information when powered off. Modern RAM includes static RAM (SRAM), dynamic RAM (DRAM), and phase-change memory (PRAM). DRAM is the predominant form of computer memory.
- *Magnetic Disks and Disk Arrays:* Magnetic disks, such as hard disk drive (HDD), are the primary component in modern storage systems. An HDD consists of one or more rigid rapidly rotating discs with magnetic heads arranged on a moving actuator arm to read and write data to the surfaces. Unlike RAM, an HDD retains its data even when powered off with much lower per-capacity cost, but the read and write operations are much slower. Because of the high expenditure of a single large capacity disk, disk arrays assemble a number of disks to achieve large capacity, high access throughput, and high availability at much lower costs.
- *Storage Class Memory:* Storage class memory refers to non-mechanical storage media, such as flash memory. In general, flash memory is used to construct solid-state drives (SSDs). Unlike HDDs, SSDs have no mechanical components, run more quietly, and have lower access times and less latency than HDDs. However, SSDs remain more expensive per unit of storage than HDDs.

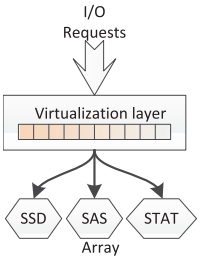


FIGURE 8. Multi-tier SSD based storage system. It consists of three components, including I/Orequest queue, virtualization layer, and array.

These devices have different performance metrics, which can be leveraged to build a scalable and high-performance big data storage subsystem. More details about storage devices development can be found in [122]. Lately, hybrid approaches [123], [124] have been proposed to build a hierarchical storage system that combines the features of SSDs and HDDs in the same unit, containing a large hard disk drive and an SSD cache to improve performance of frequently accessed data. A typical architecture of multi-tier SSD-based storage system is shown in Fig. 8, which consists of three components, i.e., I/O request queue, virtualization layer, and array [125]. Virtualization layer accepts I/O requests and dispatches them to volumes that are made up of extents stored in arrays of different device types. Current commercial SSD-based multi-tier systems from IBM, EMC, 3PAR and Compllent already gain satisfied performance. However, the major difficulty of these systems is to determine what mix of devices will perform well at minimum cost.

Second, storage infrastructure can be understood from a networking architecture perspective [126]. In this category, the storage subsystem can be organized in different ways, including, but not limited to the following.

- *Direct Attached Storage (DAS)*: DAS is a storage system that consists of a collection of data storage devices (for example, a number of hard disk drives). These devices are connected directly to a computer through a host bus adapter (HBS) with no storage network between them and the computer. DAS is a simple storage extension to an existing server.
- *Network Attached Storage (NAS)*: NAS is file-level storage that contains many hard drives arranged into logical, redundant storage containers. Compared with SAN, NAS provides both storage and a file system, and can be considered as a file server, whereas SAN is volume management utilities, through which a computer can acquire disk storage space.
- *Storage Area Network (SAN)*: SANs are dedicated networks that provide block-level storage to a group of computers. SANs can consolidate several storage devices, such as disks and disk arrays, and make them accessible to computers such that the storage devices appear to be locally attached devices.

The networking architecture of these three technologies is shown in Fig. 9. The SAN scheme possesses the most complicated architecture, depending on the specific networking devices.

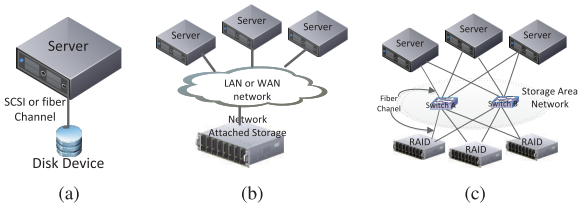


FIGURE 9. Network architecture of storage systems. It can be organized into three different architectures, including direct attached storage, network attached storage, and storage area network. (a) DAS. (b) NAS (file oriented). (c) SAN (block oriented).

Finally, existing storage system architecture has been a hot research area but might not be directly applicable to big data analytics platform. In response to the "4V" nature of the big data analytics, the storage infrastructure should be able to scale up and out and be dynamically configured to accommodate diverse applications. One promising technology to address these requirements is storage virtualization, enabled by the emerging cloud computing paradigm [127]. Storage virtualization is the amalgamation of multiple network storage devices into what appears to be a single storage device. Currently, storage virtualization [128] is achieved with SAN

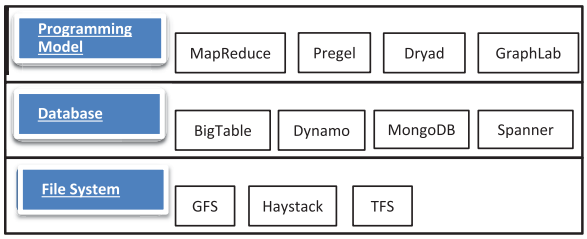


FIGURE 10. Data management technology.

or NAS architecture. SAN-based storage virtualization can gain better performance than the NAS architecture in terms of scalability, reliability, and security. However, SAN requires a professional storage infrastructure, which comes at a higher cost.

B. DATA MANAGEMENT FRAMEWORK

The data management framework concerns how to organize the information in a convenient manner for efficient processing. Data management frameworks were actively researched, even before the era of big data. In this survey, we adopt a layered view of current research efforts in this field, classifying the data management framework into three layers that consist of file systems, database technology, and programming models, as illustrated in Fig. 10. These layers are elaborated below.

1) FILE SYSTEMS

The file system is the basis of big data storage and therefore attracts great attention from both industry and academy. In this subsection, we only consider examples that are either open source or designed for enterprise use.

Google designed and implemented GFS as a scalable distributed file system [31] for large distributed data intensive applications. GFS runs on inexpensive commodity servers to provide fault tolerance and high performance to a large number of clients. It is suitable for applications with large file sizes and many more read operations than write operations. Some disadvantages of GFS, such as single point failure and poor performance for small size files, have been overcome in the successor to GFS that is known as Colossus [129]. Additionally, other companies and researchers have developed their own solutions to fulfill distinct big data storage requirements. HDFS [130] and Kosmosfs [131] are open source derivatives of GFS. Microsoft created Cosmos [132] to support its search and advertisement businesses. Facebook implemented Haystack [133] to store a massive amount of small-file photos. Two similar distributed file systems for small files, the Tao File System (TFS) [134] and FastDFS [135], have been proposed by Taobao. In summary, distributed file systems are relatively mature after a long period of large-scale commercial operation. Therefore, in this section, we emphasize the remaining two layers.

2) DATABASE TECHNOLOGIES

Database technology has gone through more than three decades of development. Various database systems have been proposed for different scales of datasets and diverse applications. Traditional relational database systems obviously cannot address the variety and scale challenges required by big data. Due to certain essential characteristics, including being schema free, supporting easy replication, possessing a simple API, eventual consistency and supporting a huge amount of data, the NoSQL database is becoming the standard to cope with big data problems. In this subsection, we mainly focus on three primary types of NoSQL databases that are organized by the data model, i.e., key-value stores, column-oriented databases, and document databases.

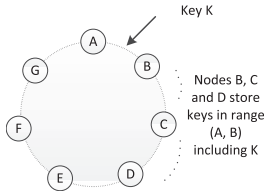


FIGURE 11. Partitioning and replication of keys in Dynamo ring [136].

a: KEY-VALUE STORES

Key-value stores have a simple data model in which data are stored as a key-value pair. Each of the keys is unique, and the clients put on or request values for each key. Key-value databases that have emerged in recent years have been heavily influenced by Amazon's Dynamo [136]. In Dynamo, data must be partitioned across a cluster of servers and replicated to multiple copies. The scalability and durability rely on two key mechanisms: partitioning and replication and object versioning.

- *Partitioning and Replication*: Dynamo's partitioning scheme relies on consistent hashing [137] to distribute the load across multiple storage hosts. In this mechanism, the output range of a hash function is treated as a fixed circular space or "ring." Each node in the system is assigned a random value within this space, which represents its "position" on the ring. Each data item identified by a key is mapped to a node by hashing the data item's key to yield the node's position on the ring. Each data item in the Dynamo system is stored in its coordinator node, and replicated at $N - 1$ successors, where N is a parameter configured per instance. As illustrated in Fig. 11, node B is a coordinator node for the key k, and the data will be replicated at nodes C and D, in addition to being stored at node B. Additionally, node D will store the keys that fall in the ranges $(A, B]$, $(B, C]$, and $(C, D]$.
- *Object Version*: Because there are multiple replications for each unique data item, Dynamo allows updates to be propagated to all replicas asynchronously to provide eventual consistency. Each update is treated as a new

and immutable version of the data. Multiple versions of an object are presented in the system concurrently, and newer versions subsume previous versions.

Other key-value stores include Voldemort [138], Redis [139], Tokyo Cabinet [140] and Tokyo Tyrant [141], Memcached [142] and MemcacheDB [143], Riak [144], and Scalaris [145]. Voldemort, Riak, tokyo Cabinet and Memecached can store data in RAM or on disk with storage add-ons. The others store in RAM and provide disk as backup, or rely on replication and recovery to eliminate the need for a backup.

b: COLUMN ORIENTED DATABASES

Column-oriented databases store and process data by column instead of by row. Both rows and columns will be split over multiple nodes to achieve scalability. The main inspiration for column-oriented databases is Google's Bigtable, which will be discussed first, followed by several derivatives.

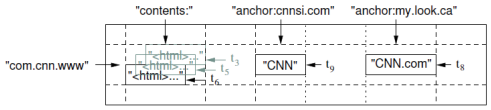


FIGURE 12. Bigtable data model [146].

- *Bigtable* [146]: The basic data structure of Bigtable is a sparse, distributed, persistent multi-dimensional sorted map. The map is indexed by a row key, a column key and a timestamp. Rows are kept in lexicographic order and are dynamically partitioned into tablets, which represent the unit of distribution and load balancing. Columns are grouped by their key prefix into sets called column families that represent the basic unit of access control. A timestamp is used to differentiate reversions of a cell value. Fig. 12 illustrates an example for storing a large collection of webpages in a single table, in which *URL*s are used as row keys and various aspects of webpages are used as column names. The contents of the webpages associated with multiple versions are stored in a single column. Bigtable's implementation consists of three major components per instance: master server, tablet server, and client library. One master server is allocated for each Bigtable runtime and is responsible for assigning tablets to tablet servers, detecting added and removed tablet servers, and distributing the workload across tablet servers. Furthermore, the master server processes changes in the Bigtable schema, such as the creation of tables and column families, and collects garbage, i.e., deleted or expired files that are stored in GFS for the particular Bigtable instance. Each tablet server manages a set of tablets, handles read and write requests for tablets, and splits tablets that have grown too large. A client library is provided for applications to interact with Bigtable instances. Bigtable depends on a number of technologies in Google's infrastructure, including GFS [31], a cluster management system, an SSTable file format and Chubby [147].
- *Cassandra* [148]: Cassandra, developed by Facebook and open-sourced in 2008, brings together the distributed system technologies from Dynamo and the data model from Bigtable. In particular, a table in Cassandra is a distributed multi-dimensional map structured across four dimensions: rows, column families, columns, and super columns. The partition and replication mechanisms of Cassandra are similar to those of Dynamo, which guarantees eventual consistency.
- *Bigtable Derivatives*: Because the Bigtable code is not available under an open source license, open source projects, such as HBase [149] and Hyper-table Hypertable [150], have emerged to implement similar systems by adopting the concepts described in the Bigtable subsection.

Column-oriented databases are similar because they are mostly patterned after Bigtable but differ in concurrency mechanisms and other features. For instance, Cassandra focuses on weak concurrency via multi-version concurrency control, whereas HBase and HyperTable focus on strong consistency via locks and logging.

c: DOCUMENT DATABASES

Document stores support more complex data structures than key-value stores. There is no strict schema to which documents must conform, which eliminates the need of schema migration efforts. In this paper, MongoDB, SimpleDB, and CouchDB are investigated as the three major representatives. The data models of all the document stores resemble the JSON [151] object. Fig. 13 shows a wiki article represented in the MongoDB [152] document format. The major differences in the document stores are in the data replication and consistency mechanisms, which are explained below.

```
{
  title: "MongoDB",
  last_editor: "172.5.123.91",
  last_modified: new Date("9/23/2013"),
  body: "MongoDb is a ..."
  categories: ["Database", "NoSQL", "Document Databases"],
  reviewed: false
}
```

FIGURE 13. MongoDB data model [146].

- *Replication and Sharding*: Replication in MongoDB is implemented using a log file on the master node that contains all high-level operations performed in the database. In a replication process, slaves ask the master for all write operations since their last synchronization and perform the operations from the log on their own local database. MongoDB supports horizontal scaling via automatic sharding to distribute data across thousands of nodes with automatic balancing of the load and automatic failover. SimpleDB simply replicates all data onto different machines in different data centers

TABLE 5. Design decision for NoSQL storage systems.

Data Model	Name	Producer	Data Storage	Concurrency Control	CAP Option	Consistency
Key-Value	Dynamo	Amazon	Plug-in	MVCC	AP	Eventually Consistent
	Voldemort	LinkeId	RAM	MVCC	AP	Eventually Consistent
	Redis	Salvatore Sanfilippo	RAM	Locks	AP	Eventually Consistent
Column	BigTable	Google	Google File Systems	Locks + stamps	CP	Eventually Consistent
	Cassandra	Facebook	Disk	MVCC	AP	Eventually Consistent
	Hbase	Apache	HDFS	Locks	CP	Eventually Consistent
	Hypertable	Hypertable	Plug-in	Locks	AP	Eventually Consistent
Document	SimpleDB	Amazon	S3 (Simple Storage Solution)	None	AP	Eventually Consistent
	MongoDB	10gen	Disk	Locks	AP	Eventually Consistent
	CouchDB	Couchbase	Disk	MVCC	AP	Eventually Consistent
Row	PNUTS	Yahoo	Disk	MVCC	AP	Timeline consistent

to ensure safety and increase performance. CouchDB uses optimistic replication to achieve scalability with no sharding mechanism currently. Each CouchDB database can be synchronized to another instance; thus, any type of replication topology can be built.

- *Consistency*: Both MongoDB and SimpleDB have no version concurrency control and no transaction management mechanisms, but they provide eventual consistency. The type of consistency of CouchDB depends on whether the master-master or master-slave configuration is used. In the former scenario, CouchDB provides eventual consistency; otherwise, CouchDB is able to guarantee strong consistency.

d: OTHER NoSQL AND HYBRID DATABASES

In addition to the aforementioned data stores, many other variant projects have been implemented to support different types of data stores, such as graph stores (Neo4j [153], DEX [154]) and PNUTS [155].

Because relational databases and NoSQL databases have their own advantages and disadvantages, one idea is to combine the two patterns to gain advanced performance. Following this trend, Google recently developed several databases to integrate the advantages of NoSQL and SQL databases, including the following.

- Megastore [156] blends the scalability of NoSQL data stores with the convenience of traditional RDBMSs to achieve both strong consistency and high availability. The design concept is that Megastore partitions the data store, replicates each partition separately, and provides full ACID semantics within partitions but only limited consistency guarantees across partitions. Megastore provides only limited traditional database features that can scale within user-tolerable latency limits and only with the semantics that the partitioning scheme can support. The data model of Megastore lies between the abstract tuple of an RDBMS and the concrete row-column storage of NoSQL. The underlying data storage of Megastore relies on Bigtable.
- Spanner [157] is the first system to distribute data on a global scale and support externally consistent distributed transactions. Unlike the versioned key-value store model in Bigtable, Spanner has evolved into a temporal multi-version database. Data are stored in schematized semi-relational tables and are versioned, and each version is automatically time stamped with its commit time. Old versions of data are subject to configurable garbage-collection policies. Applications can read data at old timestamps. In Spanner, the replication of data at a fine grain can be dynamically controlled by applications. Additionally, data are re-sharded across machines or even across data centers to balance loads and in response to failures. The salient features of Spanner are the externally consistent reads and writes and the globally consistent reads across the database at a timestamp.
- F1 [158], built on Spanner, is Google's new database for advertisement business. F1 implements rich relational database features, including a strictly enforced schema, a powerful parallel SQL query engine, general transactions, change tracking and notifications, and indexing. The store is dynamically sharded, supports transactionally consistent replication across data centers, and can handle data center outages without data loss.

e: COMPARISON OF NoSQL DATABASES

Even with so many kinds of databases, no one can be best for all workloads and scenarios, different databases make distinctive tradeoffs to optimize specific performance. Cooper et al. [159] discussed the tradeoffs faced in cloud based data management systems, including read performance versus write performance, latency versus durability, synchronous versus asynchronous replication, and data partitioning. Some other design metrics have also been argued in [160]–[162]. This paper will not attempt to argue the metrics of particular systems. Instead, Table 5 compares some salient features of the surveyed systems as follows.

- *Data Model*: This paper mainly focuses on three primary data models, i.e., key-value, column, and document. Data model in PNUTS is row oriented.
- *Data Storage*: Some systems are designed for storage in RAM with snapshots or replication to disk, while others are designed for disk storage with cache in RAM. A few systems have a pluggable back end allowing different data storage media, or they require a standardized underlying file system.
- *Concurrency Control*: There are three concurrency con-

trol mechanisms in the surveyed systems, locks, MVCC, and none. Locks mechanism allows only one user at a time to read or modify an entity (an object, document, or row). MVCC mechanism guarantees a read-consistent view of the database, but resulting in multiple conflicting versions of an entity if multiple users modify it at the same time. Some systems do not provide atomicity, allowing different users to modify different parts of the same object in parallel, and giving no guarantee as to which version of data you will get when you read.

- *CAP Option*: CAP theorem [163], [164] reveals that a shared data system can only choose at most two of three properties: consistency, availability, and tolerance to partitions. To deal with partial failures, cloud based databases also replicate data over a wide area, this essentially leaves just consistency and availability to choose. Thus, there is a tradeoff between consistency and availability. Various forms of weak consistency models [165] have been implemented to yield reasonable system availability.
- *Consistency*: Strict consistency cannot be achieved together with availability and partition tolerance according to CAP theorem. Two types of weak consistency, eventually consistency and timeline consistency, are commonly adopted. Eventual consistency means all updates can be expected to propagate through the system and the replicas will be consistent under the given long time period. Timeline consistency refers to all replicas of a given record apply all updates to the record in the same order [155].

In general,it is hard to maintain ACID guarantees in big data applications. The choice of data management tools depends on many factors including the aforementioned metrics. For instance, data model associates with the data sources; data storage devices affect the access rate. Big data storage system should find the right balance between cost, consistency and availability.

3) PROGRAMMING MODELS

Although NoSQL databases are attractive for many reasons, unlike relational database systems, they do not support declarative expression of the join operation and offer limited support of querying and analysis operations. The programming model is critical to implementing the application logics and facilitating the data analysis applications. However, it is difficult for traditional parallel models (like OpenMP [166] and MPI [167]) to implement parallel programs on a big data scale, i.e., hundreds or even thousands of commodity servers over a wide area. Many parallel programming models have been proposed to solve domain-specific applications. These efficient models improve the performance of NoSQL databases and lessen the performance gap with relational databases. NoSQL databases are already becoming the cornerstone of massive data analysis. In particular, we discuss three types of process models: the generic processing model, the graph processing model, and the stream processing model.

- *Generic Processing Model*: This type of model addresses general application problems and is used in MapReduce [13] and its variants, and in Dryad [42].
  MapReduce is a simple and powerful programming model that enables the automatic paralleling and distribution of large-scale computation applications on large clusters of commodity PCs. The computational model consists of two user-defined functions, called *Map* and *Reduce*. The MapReduce framework groups together all intermediate values associated with the same intermediate key I and passes them to the Reduce function. The Reduce function receives an intermediate key I and its set of values and merges them to generate a (typically) smaller set of values. The concise MapReduce framework only provides two opaque functions, without some of the most common operations (e.g., projection and filtering). Adding the SQL flavor on top of the MapReduce framework is an efficient way to make MapReduce easy to use for traditional database programmers skilled in SQL. Several high-level language systems, such as Google's Sawzall [168], Yahoo's Pig Latin [169], Facebook's Hive [170], and Microsoft's Scope [132], have been proposed to improve programmersąŕ productivity.
  Dryad is a general-purpose distributed execution engine for coarse-grain data parallel applications. A Dryad job is a directed acyclic graph in which each vertex is a program and edges represent data channels. Dryad runs the job by executing the vertices of this graph on a set of computers, communicating through data channels, including files, TCP pipes, and shared-memory FIFOs. The logical computation graph is automatically mapped onto physical resources in the runtime. The MapReduce programming model can be viewed as a special case of Dryad in which the graph consists of two stages: the vertices of the map stage shuffle their data to the vertices of the reduce stage. Dryad has its own high-level language called DryadLINQ [171] to generalize execution environments such as the aforementioned SQL-like languages.
- *Graph Processing Model*: A growing class of applications (e.g., social network analysis, RDF) can be expressed in terms of entities related to one another and captured using graphic models. In contrast to flow-type models, graph processing iterative by nature, and the same dataset may have to be revisited many times. We mainly consider Pregel [43] and GraphLab [172].
  Google's Pregel specializes in large-scale graph computing, such as web graph and social network analysis. The computational task is expressed as a directed graph, which consists of vertices and directed edges. Each vertex is associated with a modifiable and user-defined value. The directed edges are associated with their source vertices, and each edge consists of an alterable value and a target vertex identifier. After the initializa-

TABLE 6. Feature summary of programming models.

	MapReduce	Dryad	Pregel	GraphLab	Storm	S4
Application	general purpose parallel execution engine	general purpose parallel execution engine	large scale graph processing	large scale machine learning and data mining	distributed streaming processing	distributed streaming processing
Programming Model	Map and Reduce	directed acyclic graph	directed graph	directed Graph	directed acyclic graph	directed acyclic graph
Parallelism	concurrent execution within map and reduce phases	concurrent execution of vertices during a stage	concurrent execution over vertices within a superstep	concurrent execution of non-overlapping scopes, defined by consistency model	worker processes and executors	worker processes and executors
Data Handling	distributed file system	various storage media	distributed file system	memory or disk	memory	memory
Architecture	master-slaves	master-slaves	master-slaves	master-slaves	master-slaves	decentralized and symmetric
Fault Tolerance	node level fault tolerance	node level fault tolerance	checkpointing	checkpointing	partial fault tolerance	partial fault tolerance

tion of the graph, programs are executed as a sequence of iterations, called *supersteps*, that are separated by global synchronization points until the algorithm terminates with the output. Within each superstep, the vertices execute the same user-defined function in parallel that expresses the logic of a given algorithm. A vertex can modify its state or that of its outgoing edges, receive messages transmitted to it in the previous superstep, send messages to other vertices, or even mutate the topology of the graph. An edge has no associated computation. A vertex can deactivate itself by voting to halt. When all the vertices are simultaneously inactive and there is no message in transit, the entire program terminates. The result of a Pregel program is the set of output values by the vertices, which is frequently a directed graph isomorphic to the input.

GraphLab is another graph-processing model, which targets parallel machine learning algorithms. The GraphLab abstraction consists of three components: the data graph, the update function, and the sync operation. The data graph is a container that manages user defined data, including model parameters, algorithm state, and even statistical data. The update function is a stateless procedure that modifies the data within the scope of a vertex and schedules the future execution of update functions on other vertices. Finally, the sync operation concurrently maintains global aggregates. The key difference between Pregel and GraphLab is found in their synchronization models. Pregel has a barrier at the end of every iteration and all vertices should reach a global synchronization status after iteration, whereas GraphLab is completely asynchronous, leading to more complex vertices. GraphLab proposes three consistency models, full, edge, and vertex consistency, to allow for different levels of parallelism.

- *Stream Processing Model*: S4 [36] and Storm [35] are two distributed stream processing platforms that run on the JVM. S4 implements the actors programming model. Each keyed tuple in the data stream is treated as an event and routed with an affinity to processing elements (PEs). PEs form a directed acyclic graph and take charge of processing the events with certain keys and publishing results. Processing nodes (PNs) are the logical hosts to PEs and are responsible for listening to events and passing incoming events to the processing element container (PEN), which invokes the appropriate PEs in the appropriate order. Storm shares many feature with S4. A Storm job is also represented by a directed graph, and its fault tolerance is partial as a result of the streaming channel between vertexes. The main difference between S4 and Storm is the architecture: S4 adopts a decentralized and symmetric architecture, whereas Storm is a master-slave system such as MapReduce.

Table 6 shows a feature comparison of the programming models discussed above. First, although real-time processing is becoming more important, batch processing remains the most common data processing paradigm. Second, most of the systems adopt the graph as their programming model because the graph can express more complex tasks. Third, all the systems support concurrent execution to accelerate processing speed. Fourth, streaming processing models utilize memory as the data storage media to achieve higher access and processing rates, whereas batch-processing models employ a file system or disk to store massive data and support multiple visiting. Fifth, the architecture of these systems is typically master-slave; however, S4 adopts a decentralized architecture. Finally, the fault tolerance strategy is different for different systems. For Storm and S4, when node failure occurs, the processes on the failed node are moved to standby nodes. Pregel and GraphLab use checkpointing for fault tolerance, which is invoked at the beginning of certain iterations. MapReduce and Dryad support only node-level fault tolerance.

In addition, other research has focused on programming models for more specific tasks, such as cross joining two sets [173], iterative computation [174], [175], in-memory computation with fault-tolerance [176], incremental computation [177]–[180], and data-dependent flow control decision [181].

VII. PHASE IV: DATA ANALYSIS

The last and most important stage of the big data value chain is data analysis, the goal of which is to extract useful values, suggest conclusions and/or support decision-making. First, we discuss the purpose and classification metric of data analytics. Second, we review the application evolution for various data sources and summarize the six most relevant areas. Finally, we introduce several common methods that play fundamental roles in data analytics.

A. PURPOSE AND CATEGORIES

Data analytics addresses information obtained through observation, measurement, or experiments about a phenomenon of interest. The aim of data analytics is to extract as much information as possible that is pertinent to the subject under consideration. The nature of the subject and the purpose may vary greatly. The following lists only a few potential purposes:

- To extrapolate and interpret the data and determine how to use it,
- To check whether the data are legitimate,
- To give advice and assist decision-making,
- To diagnose and infer reasons for fault, and
- To predict what will occur in the future.

Because of the great diversity of statistical data, the methods of analytics and the manner of application differ significantly. We can classify data into several types according to the following criteria: qualitative or quantitative with the property of the observation or measurement and univariate or multivariate according to the parameter count. Additionally, there have been several attempts to summarize the domain-related algorithms. Manimon et al. [182] presented a taxonomy of data-mining paradigms. In this taxonomy, data mining algorithms can be categorized as descriptive, predictive, and verifying. Bhatt et al. [183] categorized multimedia analytics approaches into feature extraction, transformation, representation, and statistical data mining. However, little effort has been made to classify the entire field of big data analytics. Blackett et al. [184] classified data analytics into three levels according to the depth of analysis: descriptive analytics, predictive analytics, and prescriptive analytics.

- *Descriptive Analytics*: exploits historical data to describe what occurred. For instance, a regression may be used to find simple trends in the datasets, visualization presents data in a meaningful fashion, and data modeling is used to collect, store and cut the data in an efficient way. Descriptive analytics is typically associated with business intelligence or visibility systems.
- *Predictive Analytics*: focuses on predicting future probabilities and trends. For example, predictive modeling uses statistical techniques such as linear and logistic regression to understand trends and predict future outcomes, and data mining extracts patterns to provide insight and forecasts.
- *Prescriptive Analytics*: addresses decision making and efficiency. For example, simulation is used to analyze complex systems to gain insight into system behavior and identify issues and optimization techniques are used to find optimal solutions under given constraints.

B. APPLICATION EVOLUTION

More recently, big data analytics has been proposed to describe the advanced analysis methods or mechanisms for massive data. In fact, data-driven applications have been emerging for the past few decades. For example, business intelligence became a popular term in business communities early in the 1990s and data mining-based web search engines arose in the 2000s. In the following, we disclose the evolution of data analysis by presenting high impact and promising applications from typical big data domains during different time periods.

1) BUSINESS APPLICATION EVOLUTION

The earliest business data were structured data, which are collected by companies and stored in relational database management systems. The analysis techniques used in these systems, which were popularized in the 1990s, are commonly intuitive and simple. Gartner [185] summarized the most common business intelligence methods, including reporting, dashboards, ad hoc queries, search-based business intelligence, online transaction processing, interactive visualization, scorecards, predictive modeling, and data mining. Since the early 2000s, the Internet and the web offered a unique opportunity for organizations to present their businesses online and interact with customers directly. An immense amount of products and customer information, including clickstream data logs and user behavior, can be gathered from the web. Using various text and web mining techniques in analysis, product placement optimization, customer transaction analysis, product recommendations, and market structure analysis can be undertaken. As reported [186], the number of mobile phones and tablets surpassed the number of laptops and PCs for the first time in 2011. Mobile phones and the Internet of Things created additional innovative applications with distinctive features, such as location awareness, person-centered operation, and context relevance.

2) NETWORK APPLICATION EVOLUTION

The early network mainly provided email and website service. Thus, text analysis, data mining and webpage analysis techniques were widely adopted to mine email content, construct search engines, etc. Currently, almost all applications, regardless of their purpose or domains, run on a network. Network data has dominated the majority of global data volumes. The web is a growing universe with interlinked webpages that is teeming with diverse types of data, including text, images, videos, photos, and interactive content. Various advanced technologies for semi-structured or unstructured data have been proposed. For example, image analysis can extract meaningful information from photos and multimedia analysis techniques can automate video surveillance systems

in commercial, law enforcement, and military applications. After 2004, online social media, such as forums, groups, blogs, social network sites, and multimedia sharing sites, offer attractive opportunities for users to create, upload, and share an abundant amount of user-generated content. Mining everyday events, celebrity chatter, and socio-political sentiments expressed in these media from a diverse customer population provides timely feedback and opinions.

3) SCIENTIFIC APPLICATION EVOLUTION

Many areas of scientific research are reaping a huge volume of data from high throughput sensors and instruments, from the fields of astrophysics and oceanography to genomics and environmental research. The National Science Foundation (NSF) recently announced the *BIGDATA* program solicitation [187] to facilitate information sharing and data analytics. Several scientific research disciplines have previously developed massive data platform and harvested the resulting benefits. For example, in biology, iPlant [188] is using cyberinfrastructure, physical computing resources, a collaborative environment, virtual machine resources and interoperable analysis software and data services to support a community of researchers, educators, and students working to enrich all plant sciences. The iPlant data set is diverse and includes canonical or reference data, experimental data, simulation and model data, observational data, and other derived data.

From the above description, we can classify data analytics research into six critical technical areas: structured data analysis, text analytics, web analytics, multimedia analytics, network analytics, and mobile analytics. This classification is intended to highlight the data characteristics; however, a few of these areas may leverage similar underlying technologies. Our aim is to provide an understanding of the primary problems and techniques in the data analytics field, although being exhaustive is difficult because of the extraordinarily broad spectrum of data analytics.

C. COMMON METHODS

Although the purpose and application domains differ, some common methods are useful for almost all of the analysis. Below, we discuss three types of data analysis methods.

- *Data visualization*: is closely related to information graphics and information visualization. The goal of data visualization is to communicate information clearly and effectively through graphical means [189]. In general, charts and maps help people understand information easily and quickly. However, as the data volume grows to the level of big data, traditional spreadsheets cannot handle the enormous volume of data. Visualization for big data has become an active research area because it can assist in algorithm design, software development, and customer engagement. Friedman [190] and Frits [191] summarized this field from the information representation and computer science perspectives, respectively.
- *Statistical analysis*: is based on statistical theory, which is a branch of applied mathematics. Within statistical theory, randomness and uncertainty are modeled by probability theory. Statistical analysis can serve two purposes for large data sets: description and inference. Descriptive statistical analysis can summarize or describe a collection of data, whereas inferential statistical analysis can be used to draw inferences about the process. More complex multivariate statistical analysis [192] uses analytical techniques such as aggression , factor analysis, clustering, and discriminant analysis.
- *Data mining*: is the computational process of discovering patterns in large data sets. Various data mining algorithms have been developed in the artificial intelligence, machine learning, pattern recognition, statistics, and database communities. During the 2006 IEEE International Conference on Data Mining (ICDM), the ten most influential data mining algorithms were identified based on rigorous election [193]. In ranked order, these algorithms are C4.5, k-means, SVM (Support Vector Machine), a priori, EM (Expectation Maximization), PageRank, AdaBoost, kNN, Naive Bayes, and CART. These ten algorithms cover classification, clustering, regression, statistical learning, association analysis and link mining, which are all among the most important topics in research on data mining. In addition, other advanced algorithms, such as neural network and genetic algorithms, are useful for data mining in different applications.

VIII. CASES IN POINT OF BIG DATA ANALYTICS

According to the application evolution depicted in the previous section, we discuss six types of big data application, organized by data type: structured data analytics, text analytics, web analytics, multimedia analytics, network analytics, and mobile analytics.

A. STRUCTURED DATA ANALYTICS

A large quantity of structured data is generated in the business and scientific research fields. Management of these structured data relies on the mature RDBMS, data warehousing, OLAP, and BPM [46]. Data analytics is largely grounded in data mining and statistical analysis, as described above. These two fields have been thoroughly studied in the past three decades. Recently, deep learning, a set of machine-learning methods based on learning representations, is becoming an active research field. Most current machine-learning algorithms depend on human-designed representations and input features, which is a complex task for various applications. Deep-learning algorithms incorporate representation learning and learn multiple levels of representation of increasing complexity/abstraction [194].

In addition, many algorithms have been successfully applied to emerging applications. Statistical machine learning, based on precise mathematical models and powerful algorithms, has already been applied in anomaly detec-

tion [195] and energy control [196]. Utilizing data characteristics, temporal and spatial mining can extract knowledge structures represented in models and patterns for high-speed data streams and sensor data [197]. Driven by privacy concerns in e-commerce, e-government, and healthcare applications, privacy-preserving data mining [198] is becoming an active research area. Over the past decade, because of the growing availability of event data and process discovery and conformance-checking techniques, process mining [199] has emerged as a new research field that focuses on using event data to analyze processes.

B. TEXT ANALYTICS

Text is one of the most common forms of stored information and includes e-mail communication, corporate documents, webpages, and social media content. Hence, text analytics is believed to have higher commercial potential than structured data mining. In general, text analytics, also known as text mining, refers to the process of extracting useful information and knowledge from unstructured text. Text mining is an interdisciplinary field at the intersection of information retrieval, machine learning, statistics, computational linguistics, and, in particular, data mining. Most text mining systems are based on text representation and natural language processing (NLP), with emphasis on the latter.

Document presentation and query processing are the foundations for developing the vector space model, Boolean retrieval model, and probabilistic retrieval model [200]. These models in turn have become the basis of search engines. Since the early 1990s, search engines have evolved into mature commercial systems, commonly performing distributed crawling, efficient inverted indexing, inlink-based page ranking, and search log analytics.

NLP techniques can enhance the available information about text terms, allowing computers to analyze, understand, and even generate text. The following approaches are frequently applied: lexical acquisition, word sense disambiguation, part-of-speech tagging, and probabilistic context-free grammars [201]. Based on these approaches, several technologies have been developed for text mining, including information extraction, topic modeling, summarization, categorization, clustering, question answering, and opinion mining. Information extraction refers to the automatic extraction of specific types of structured information from text. As a subtask of information extraction, named-entity recognition (NER) aims to identify atomic entities in text that fall into predefined categories, such as person, location, and organization. NER has recently been successfully adopted for news analysis [202] and biomedical applications [203]. Topic models are based upon the idea that documents are mixtures of topics, in which a topic is a probability distribution over words. A topic model is a generative model for documents, which specifies a probabilistic procedure by which documents can be generated. A variety of probabilistic topic models have been used to analyze the content of documents and the meaning of words [204]. Text summarization generates an abridged summary or abstract from a single or multiple input text documents and can be divided into extractive summarization and abstractive summarization [205]. Extractive summarization selects important sentences and paragraphs from the original document and concatenates them into a shorter form, whereas abstractive summarization understands the original text and retells it in fewer words, based on linguistic methods. The purpose of text categorization is to identify the main themes in a document by placing the document into a predefined topic or set of topics. Graph representation and graph mining-based text categorization have also been researched recently [206]. Text clustering is used to group similar documents and differs from categorization in that documents are clustered as they are found instead of using predefined topics. In text clustering, documents can appear in multiple subtopics. Some clustering algorithms from the data mining community are commonly used to calculate similarity. However, research has shown that structural relationship information can be leveraged to enhance the clustering result in Wikipedia [207]. A question answering system is primarily designed to determine how to find the best answer to a given question and involves various techniques for question analysis, source retrieval, answer extraction, and answer presentation [208]. Question answering systems can be applied in many areas, including education, websites, health, and defense. Opinion mining, which is similar to sentiment analysis, refers to the computational techniques for extracting, classifying, understanding, and assessing the opinions expressed in news, commentaries, and other user-generated contents. It provides exciting opportunities for understanding the opinions of the general public and customers regarding social events, political movements, company strategies, marketing campaigns, and product preferences [209].

C. WEB ANALYTICS

Over the past decade, we have witnessed an explosive growth of webpages, whose analysis has emerged as an active field. Web analytics aims to retrieve, extract, and evaluate information for knowledge discovery from web documents and services automatically. This field is built on several research areas, including databases, information retrieval, NLP, and text mining. We can categorize web analytics into three areas of interest based on which part of the web is mined: web content mining, web structure mining, and web usage mining [210].

Web content mining is the discovery of useful information or knowledge from website content. However, web content may involve several types of data, such as text, image, audio, video, symbolic, metadata, and hyperlinks. Recent research on mining image, audio, and video is termed multimedia analytics, which will be investigated in the following section. Because most of the web content data are unstructured text data, much of the research effort is centered on text and hypertext content. Text mining is a well-developed subject, as described above. Hypertext mining involves mining semi-

structured HTML pages that have hyperlinks. Supervised learning or classification plays a key role in hypertext mining, such as in email management, newsgroup management, and maintaining web directories [211]. Web content mining commonly takes one of two approaches: information retrieval or database. The information retrieval approach mainly aims to assist in information finding or in filtering information to the users based on either inferred or solicited user profiles. The database approach models the data on the web and integrates them so that more sophisticated queries other than the keyword-based searches might be performed.

Web structure mining is the discovery of the model underlying link structures on the web. Here, structure represents the graph of links in a site or between sites. The model is based on the topology of the hyperlink with or without link description. This model reveals the similarities and relationships among different websites and can be used to categorize websites. The Page Rank [212] and CLEVER [213] methods exploit this model to find webpages. Focused crawling [214] is another example that successfully utilizes this model. The goal of a focused crawler is to selectively seek out websites that are related to a predefined set of topics. Rather than collecting and indexing all accessible web documents, a focused crawler analyzes its crawl boundary to find links that are likely to be most relevant for the crawl and avoids irrelevant regions of the web, which saves significant hardware and network resources and helps to keep the crawl more up-to-date.

Web usage mining refers to mining secondary data generated by web sessions or behaviors. Web usage mining differs from web content mining and web structure mining, which utilize the real or primary data on the web. The web usage data includes the data from web server access logs, proxy server logs, browser logs, user profiles, registration data, user sessions or transactions, cookies, user queries, bookmark data, mouse clicks and scrolls, and any other data generated by the interaction of users and the web. As web services and web 2.0 systems are becoming more mature and increasing in popularity, web usage data are becoming more diversified. Web usage mining plays an important role in personalizing space, e-commerce, web privacy/security, and several other emerging areas. For example, collaborative recommendation systems allow personalization for e-commerce by exploiting similarities and dissimilarities in user preferences [215].

D. MULTIMEDIA ANALYTICS

Recently, multimedia data, including image, audio, and video, has grown at a phenomenal rate and is almost ubiquitous. Multimedia content analytics refers to extracting interesting knowledge and understanding the semantics captured in multimedia data. Because multimedia data are diverse and more information-rich than the simple structured data and text data in most of the domains, information extraction involves overcoming the semantic gap of multimedia data. The research in multimedia analytics covers a wide spectrum of subjects, including multimedia summarization, multimedia annotation, multimedia indexing and retrieval, multimedia recommendation, and multimedia event detection, to name only a few recent areas of focus.

Audio summarization can be performed by simply extracting salient words or sentences from the original data or by synthesizing new representations. Video summarization involves synthesizing the most important or representative sequences of the video content and can be static or dynamic. Static video summarization methods use a sequence of key frames or context-sensitive key frames to represent video. These methods are simple and have previously been commercialized in Yahoo, Alta Vista and Google; however, they engender a poor playback experience. Dynamic video summarization methods utilize a sequence of video segments to represent the video and employ low-level video features and perform an extra smoothing step to make the final summary look more natural. [216] proposed a topic-oriented multimedia summarization system that is capable of generating text-based recounting for videos that can be viewed at one time.

Multimedia annotation refers to assigning images and videos a set of labels that describe their content at syntactic and semantic levels. With the help of these labels, the management, summarization, and retrieval of multimedia content can be accomplished easily. Because manual annotation is time consuming and requires intensive labor costs, automatic multimedia annotation with no human interference has attracted substantial research interest. The main challenge of automatic multimedia annotation lies in the semantic gap, namely the gap between low-level features and annotations. Although significant progress has been made, the performance of current automatic annotation methods remains far from satisfactory. Emerging research efforts aim to simultaneously explore humans and the computer for multimedia annotation [217].

Multimedia indexing and retrieval concerns the description, storage, and organization of multimedia information to help people find multimedia resources conveniently and quickly [218]. A general video retrieval framework consists of four steps: structure analysis; feature extraction; data mining, classification and annotation; and query and retrieval. Structure analysis aims to segment a video into a number of structural elements with semantic content, using shot boundary detection, key frame extraction, and scene segmentation. Upon obtaining the structure analysis results, the second step is to extract features-consisting mainly of features of the key frames, objects, text, and motion for further mining [219]–[221]. This step is the basis of video indexing and retrieval. Using the extracted features, the goal of data mining, classification, and annotation is to find patterns of video content and assign the video into predefined categories to generate video indices. When a query is received, a similarity measure method is employed to search for the candidate videos. The retrieval results are optimized by relevance feedback.

The objective of multimedia recommendation is to suggest specific multimedia contents for a user based on user preferences, which has been proven as an effective scheme to provide high-quality personalization. Most current rec-

ommendation systems are either content based or collaborative filtering based. Content-based approaches identify common features of user interest, and recommend to the user other content that shares similar features. These approaches fully rely on content similarity measures and suffer from the problems of limited content analysis and over-specification. Collaborative filtering-based approaches identify the group of people who share common interests and recommend content based on other group members' behavior [222]. Hybrid approaches [223] exploit the benefits of both collaborative filtering and content-based methods to improve the quality of recommendations.

NIST defines multimedia event detection [224] as detecting the occurrence of an event within a video clip based on an event kit that contains a text description about the concept and video examples. The research on video event detection remains in its infancy. Most current research on event detection is limited to sports or news events, repetitive patterns events such as running or unusual events in surveillance videos. Ma et al. [225] proposed a novel algorithm for ad hoc multimedia event detection, which addresses a limited number of positive training examples.

E. NETWORK ANALYTICS

Because of the rapid growth of online social networks, network analysis has evolved from earlier bibliometric analysis [226] and sociology network analysis [227] to the emerging social network analysis of the early 2000s. Typically, social networks contain a tremendous amount of linkage and content data, where linkage data are essentially the graph structure, representing communications between entities and the content data contains text, images, and other multimedia data in the networks. Obviously, the richness of social network data provides unprecedented challenges and opportunities for data analytics. From the data-centric view, there are two primary research directions in the context of social networks: linkage-based structural analysis and content-based analysis [228].

Linkage-based structural analysis focuses on areas of link prediction, community detection, social network evolution, and social influence analysis, to name a few. Social networks can be visualized as graphs, in which a vertex corresponds to a person, and an edge represents certain associations between the corresponding persons. Because social networks are dynamic, new vertices and edges are added to the graph over time. Link prediction aims to forecast the likelihood of a future association between two nodes. There are a variety of techniques for link prediction, which can be categorized into feature-based classifications, probabilistic approaches and linear algebraic approaches. Feature-based classification methods choose a set of features for vertex-pairs and employ current link information to train a binary classifier to predict future links [229]. Probabilistic approaches model the joint probability among the vertices in a social network [230]. Linear algebraic approaches calculate the similarities between the nodes using rank-reduced similarity matrices [231]. Community refers to a sub-graph structure within which vertices have a higher density of edges, whereas vertices between sub-graphs have a lower density. Many methods have been proposed and compared for community detection [232], most of which are topology-based and rely on an objective function that captures the concept of the community structure. Du et al. [233] utilized the nature of overlapping communities in the real world and proposed more efficient community detection in large-scale social networks. Research on social the evolution of networks aims to find laws and derive models to explain network evolution. Several empirical studies [234]–[236] have found that proximity bias, geographic constraints, and certain other factors play an important role in the evolution of social networks, and several generative models [237] have been proposed to assist the network and system design. Social influence results when the behavior of individuals is affected by others within the network. The strength of social influence [238] depends on many factors, including relationships between persons, network distance, temporal effects, and characteristics of networks and individuals. Qualitatively and quantitatively measuring the influence [239] of one person on others can greatly benefit many applications, including marketing, advertising, and recommendation. In general, the performance of linkage-based structure analysis can be improved when the content proliferating over the social networks is considered.

Because of the revolutionary development of Web 2.0 technology, user-generated content is exploding on social networks. The term *social media* is employed to name such user-generated content, including blogs, microblogs, photo and video sharing, social book marketing, social networking sites, social news and wikis. Social media content contains text, multimedia, locations and comments. Almost every research topic on structured data analytics, text analytics, and multimedia analytics can be translated to social media analytics. However, social media analytics face certain unprecedented challenges. First, there are tremendous and ever-growing social media data, and we must analyze them within a reasonable time constraint. Second, social media data contains many noisy data. For example, spam blogs are abundant in the blogosphere, as are trivial tweets in Twitter. Third, social networks are dynamic, ever-changing and updated rapidly. In brief, social media is closely adhered to social networks, the analysis of which is inevitably affected by social network dynamics. Social media analytics refers to text analytics and multimedia analytics in the context of the social network, specifically, the social and network structure characteristics. Research on social media analytics remains in its infancy. Applications of text analytics in social networks include key word searches, classifications, clustering, and transfer learning in heterogeneous networks. Keyword searching utilizes both content and linkage behaviors [240]. Classification assumes that some nodes in social networks have labels and that these labeled nodes can be used for classification [241]. Clustering is accomplished by determin-

ing sets of nodes with similar content [242]. Because social networks contain a large amount of linked information among different types of objects, such as articles, tags, images, and videos, transfer learning in heterogeneous networks aims to transfer information knowledge across links [243]. In social networks, multimedia datasets are structured and incorporate rich information such as semantic ontology, social interaction, community media, geographical maps, and multimedia comments. Research on structured multimedia analytics in social networks is also called multimedia information networks. The link structures of multimedia information networks are primarily logical and play a vital role in multimedia information networks. There are four categories of logical link structures in multimedia information networks: semantic ontologies, community media, personal photograph albums, and geographical locations [228]. Based on the logical link structures, we can further improve the results of the retrieval system [244], the recommendation system [245], collaborative tagging [246] and other applications [247], [248].

F. MOBILE ANALYTICS

With the rapid growth of mobile computing [249]–[251], more mobile terminals (like mobile phones, sensors, RFID) and applications are deployed globally. Mobile data traffic reached 885 PBs per month at the end of 2012 [252]. The huge volume of applications and data leads to the emergence of mobile analytics; however, mobile analytics faces challenges caused by the inherent characteristics of mobile data, such as mobile awareness, activity sensitivity, noisiness, and redundancy richness. Currently, mobile analytics is far from mature; thus, we investigate only some of the latest and representative analysis applications.

RFID allows a sensor to read a unique product identification code (EPC) associated with a tag from a distance [253]. Tags can be used to identify, locate, track and monitor physical objects cost effectively. Currently, RFID is widely adopted in inventory management and logistics. However, RFID data poses many challenges for data analysis: (i) RFID data are inherently noisy and redundant; (ii) RFID data are temporal, streaming, high volume and must be processed on the fly. By mining the semantics of RFID, including location, aggregation, and temporal information, we can infer certain primitive events to track objects and monitor the system status. Furthermore, we can devise application logic as complicated events and then detect the events to accomplish more advanced business applications. A shoplifting example that uses high-level complex events is discussed in [254].

Recent advances in wireless sensors, mobile technologies, and streaming processing have led to the deployment of body sensor networks for real-time monitoring of an individualąŕs health. In general, healthcare data come from heterogeneous sensors with distinct characteristics, such as diverse attributes, spatial-temporal relationships, and physiological features. In addition, healthcare information carries privacy and security concerns with it. Garg et al. [255] presented a multi-modal analysis mechanism for the raw data stream to monitor health status in real time. With only highly aggregated health-related features available, Park et al. [256] sought a better utilization of such aggregated information to augment individual-level data. Aggregated statistics over certain partitions were utilized to identify clusters and impute features that were observed as more aggregated values. The imputed features were further used in predictive modeling to improve performance.

Under the metric discussed above, the vast majority of analysis belongs to either descriptive analytics or predictive analytics. Due to the complexity of classification, we only summarized data analysis approaches from the data life-cycle perspective, covering data sources, data characteristics, and approaches, as illustrated in Table 7.

IX. HADOOP FRAMEWORK AND APPLICATIONS

Because of the great success of Google's distributed file system and the MapReduce computation model in handling massive data processing, its clone, Hadoop, has attracted substantial attention from both industry and scholars alike. In fact, Hadoop has long been the mainstay of the big data movement. Apache Hadoop is an open-source software framework that supports massive data storage and processing. Instead of relying on expensive, proprietary hardware to store and process data, Hadoop enables distributed processing of large amounts of data on large clusters of commodity servers. Hadoop has many advantages, and the following features make Hadoop particularly suitable for big data management and analysis:

- *Scalability*: Hadoop allows hardware infrastructure to be scaled up and down with no need to change data formats. The system will automatically redistribute data and computation jobs to accommodate hardware changes.
- *Cost Efficiency*: Hadoop brings massively parallel computation to commodity servers, leading to a sizeable decrease in cost per terabyte of storage, which makes massively parallel computation affordable for the ever-growing volume of big data.
- *Flexibility*: Hadoop is free of schema and able to absorb any type of data from any number of sources. Moreover, different types of data from multiple sources can be aggregated in Hadoop for further analysis. Thus, many challenges of big data can be addressed and solved.
- *Fault tolerance*: Missing data and computation failures are common in big data analytics. Hadoop can recover the data and computation failures caused by node breakdown or network congestion.

In this section, we describe the core architecture of the Hadoop software library and introduce some cases both from industry and the academy.

A. HADOOP SOFTWARE STACKS

The Apache Hadoop software library is a massive computing framework consisting of several modules, including HDFS, Hadoop MapReduce, HBase, and Chukwa. These modules

TABLE 7. Taxonomy of big data analytics.

Analysis Domains	Sources	Characteristics	Approaches
Structured data analytics	Customer transaction records Scientific experiments data	Structured records Less volume and real time	Data mining [193] Statistical analysis [192]
Text analytics	Logs Email Corporate documents Government Rules and Regulations Text content of webpages Citizen feedback and comments	Unstructured Rich textual Context Semantic Language dependent	Document presentation [200] NLP [201] Information extraction [202] [203] Topic model [204] Summarization [205] Categorization [206] Clustering [207] Question answering [208] Opinion mining [209]
Web analytics	Various webpages	Integration of text and hyperlink Symbolic Metadata	Web content mining [211] Web structure mining [212] [213] [214] Web usage mining [215]
Multimedia analytics	Corporation produced multimedia User generated multimedia Surveillance Health and patient media	Image, audio, video Massive Redundancy Semantic gap	Summarization [216] Annotation [217] Indexing and retrieval [218] Recommendation [222] [223] Event detection [225]
Network analytics	Bibliometric Sociology network Social networks	Rich content Social relationship Noisy & Redundancy Fast evolution	Link prediction [229] [230] [231] Community detection [232] [233] Network evolution [234] [235] [236] [237] Influence analysis [238] [239] Key words search [240] Classification [241] Clustering [242] Transfer learning [243]
Mobile analytics	Mobile apps Sensors RFID	Location based Person specific Fragmented information	Monitoring [254] [255] [256] Location based mining

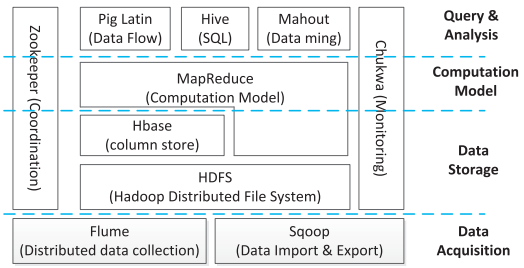


FIGURE 14. A hierarchical architecture of Hadoop core software library, covering the main function of big data value chain, including data import, data storage and data processing.

fulfill parts of the functions of a big data value chain and can be orchestrated into powerful solutions for batch-type big data applications. The layered architecture of the core library is shown in Fig. 14. We will introduce different modules from the bottom-up in examining the structure of the big data value chain.

Apache Flume and Sqoop are two data integration tools that can accomplish the data acquisition of the big data value chain. Flume is a distributed system that efficiently collects, aggregates, and transfers large amounts of log data from disparate sources to a centralized store. Sqoop allows easy import and export of data among structured data stores and Hadoop.

Hadoop HDFS and HBase are responsible for data storage. HDFS is a distributed file system developed to run on commodity hardware that references the GFS design. HDFS is the primary data storage substrate of Hadoop applications. An HDFS cluster consists of a single *NameNode* that manages the file system metadata, and collections of *DataNodes* that store the actual data. A file is split into one or more blocks, and these blocks are stored in a set of DataNodes. Each block has several replications distributed in different DataNodes to prevent missing data. Apache HBase is a column-oriented store modeled after Google's Bigtable. Thus, Apache HBase provides Bigtable-like capabilities as discussed in the last section VI above on top of HDFS. HBase can serve both as the input and the output for MapReduce jobs run in Hadoop and may be accessed through Java API, REST, Avor or Thrift APIs.

Hadoop MapReduce is the computation core for massive data analysis and is also modeled after Google's MapReduce. The MapReduce framework consists of a single master *JobTracker* and one slave *TaskTracker* per cluster node. The master is responsible for scheduling jobs for the slaves, monitoring them and re-executing the failed tasks. The slaves execute the tasks as directed by the master. The MapReduce framework and HDFS run on the same set of nodes, which allows tasks to be scheduled on the nodes in which data are already present.

Pig Latin and Hive are two SQL-like high-level declarative languages that express large data set analysis tasks in MapReduce programs. Pig Latin is suitable for data flow tasks and can produce sequences of MapReduce programs, whereas Hive facilitates easy data summarization and ad hoc queries. Mahout is a data mining library implemented on

TABLE 8. Hadoop module summarization.

Function	Module	Description
Data Acquisition	Flume	Data collection from disparate sources to a centralized store
	Sqoop	Data import and export between structured stores and Hadoop
Data Storage	HDFS	Distributed file system
	Hbase	Column-based data store
Computation	MapReduce	group-aggregation computation framework
Query & Analysis	Pig Latin	SQL-like language for data flow tasks
	Hive	SQL-like language for data query
	Mahout	data mining library
Management	Zokeeper	service configuration, synchronization, etc.
	Chukwa	system monitoring

top of Hadoop that uses the MapReduce paradigm. Mahout contains many core algorithms for clustering, classification, and batch-based collaborative filtering.

Zookeeper and Chukwa are used to manage and monitor distributed applications that run on Hadoop. Specifically, Zookeeper is a centralized service for maintaining configuration, naming, providing distributed synchronization, and providing group services. Chukwa is responsible for monitoring the system status and can display, monitor, and analyze the data collected.

Table 8 presents a quick summary of the function classification of Hadoop modules, covering most parts of the big data value chain. Under this classification, Flume and Sqoop fulfill the function of data acquisition, HDFS and Hbase are responsible for data storage, MapReduce, Pig Latin, Hive, and Mahout perform data processing and query functions, and ZooKeeper and Chukwa coordinate different modules being run in the big data platform.

B. DEPLOYMENT

Hadoop is now widely adopted industrially for various applications, including spam filtering, web search, click stream analysis, and social network recommendation. Moreover, much academic research is built upon Hadoop. In the following, we survey some representative cases.

As announced in June 2012, Yahoo! runs Hadoop on 42,000 servers in four data centers to support Yahoo! products and projects, such as Yahoo! search and spam filtering. Its largest Hadoop cluster holds 4,000 nodes but will increase to 10,000 with the release of Apache Hadoop 2.0. In the same month, Facebook announced that their Hadoop cluster processed 100 PB data, and this volume grew by roughly half a PB per day in November 2012. Some notable organizations that use Hadoop to run large distributed computations can be found in [10]. In addition, there are a number of companies offering commercial implementation and/or providing support for Hadoop, including Cloudera, IBM, MapR, EMC, and Oracle.

The exponential increase of genomic data and the dramatic drop in sequencing costs have changed the landscape of biological and medical research. Scientific analysis is increasingly data driven. Gunarathne et al. [257] used a cloud infrastructure, Amazon AWS and Microsoft Azure, in addition to data processing frameworks-Hadoop and Microsoft DryadLINQ, to implement two parallel biomedical applications: 1) the assembly of genome fragments and 2) dimension reduction in the analysis of chemical structures. The data set in the latter application is 166-dimensional and features 26 million data points. A comparative study of the two frameworks was conducted based on performance, efficiency, cost and usability. The study suggests that loosely coupled science applications will increasingly be implemented on clouds and that using the MapReduce framework will offer a convenient user interfaces with little overhead.

C. IMPROVEMENTS

Despite many advantages, Hadoop still lacks certain features found in DBMS, which is over 40 years old. For example, because Hadoop has no schema and no index, it must parse each item when reading the input and transform the input into data objects, which leads to performance degradation. Hadoop provides a single fixed dataflow; nevertheless, many complex algorithms are hard to implement with only Map and Reduce in a job. The following represent several approaches that are currently used to improve the pitfalls of the Hadoop framework:

- *Flexible Data Flow:* Many algorithms cannot directly map into MapReduce functions, including loop-type algorithms that require state information for execution and termination. Researchers have attempted to extend Hadoop to support flexible data flow; HaLoop [258] and Twister [259] are such systems that support loop programs in MapReduce.
- *Blocking Operators:* The Map and Reduce functions are blocking operations, i.e., a job cannot move forward to the next stage until all tasks are completed at the original stage. This property causes performance degradation and makes Hadoop unsuitable for on-line processing. Logothetis et al. [260] built MapReduce abstraction onto their distributed engine for ad hoc data processing. MapReduce Online [261] is devised to support online aggregation and continuous queries. Li et al. [262] and Jiang et al. [263] utilized hash tables for better performance and incremental processing.
- *I/O Optimization:* Some approaches leverage index structures or data compression to reduce the I/O cost in Hadoop. Hadoop++ [264] provides an index-structured file format that improves the I/O cost. HadoopDB [265] leverages DBMS as storage in each node to benefit from the DB indexes.
- *Scheduling:* The Hadoop scheduler implements a simple heuristic scheduling strategy that compares the progress of each task to the average progress to determine re-execution tasks. This method is not suitable for heterogeneous environments. Longest Approximation Time

to End (LATE) scheduling has been devised to improve the response time of Hadoop in heterogeneous environments. In a multi-user environment in which users simultaneously execute their jobs in a cluster, Hadoop implements two scheduling schemes: fair scheduling and capacity scheduling. These two methods lead to poor resource utilization. Many researchers are working to improve the scheduling policies in Hadoop, such as the delay scheduler [266], dynamic proportional scheduler [267], deadline constraint scheduler [268], and resource-aware scheduler [269].

- *Joins:* MapReduce is designed for processing a single input. The extension of the supporting join operator allows Hadoop to dispose multiple inputs. Join methods can be roughly classified into two groups: Map-side join [270] and Reduce-side join [271].
- *Performance Tuning:* Hadoop provides a general framework to support a variety of applications, but the default configuration scheme does not guarantee that all the applications run the best. Babu et al. [272] proposed an automatic tuning approach to find optimal system parameters for the given input data. Jahani et al. [273] presented a static analysis method for the automatic optimization of a single MapReduce job.
- *Energy Optimization:* A Hadoop cluster commonly consists of a large collection of commodity servers, which consume a substantial amount of energy. An energy efficient method for controlling nodes in a Hadoop cluster must be devised. The Covering-Set approach [274] designates certain nodes to host at least a replica of each data block, and other nodes are powered off during low-utilization periods. The All-In strategy [275] saves energy by powering off all nodes until the job queue exceeds a predetermined threshold.

Please refer to [276] and [277] for more details on this topic.

Hadoop is designed for batch-type application. In many real-time applications, Storm [35] is a good candidate for processing unbounded streams of data. Storm can be used for real-time analytics, online machine learning, continuous computation, and distributed RPC. Recently, Twitter disclosed their open project, called Summingbird [278], which integrates Hadoop and Storm.

X. BIG DATA SYSTEM BENCHMARK

A. CHALLENGES

The TPC (Transaction Processing Performance Council) series of benchmarks have greatly accelerated the development and commercialization of traditional relational databases. As big data systems mature, scholarly and industrial researchers try to create TPC-like benchmarks to evaluate and compare the performance of these systems. However, to date, there are no standard benchmarks available. The unique characteristics of big data systems present the following challenges for benchmark efforts [279]:

- *System Complexity*: Big data systems are commonly the organic composition of multiple modules or components. These modules have different functions and are coupled together. Modeling the entire system and refining a unified framework suitable for every module is not straightforward.
- *Application Variety*: A well-defined benchmark must reflect the representative characteristics of big data systems, such as the skew of the data types, the application access pattern, and the performance requirements. Because of the diversity of big data applications, extracting the salient features is complicated.
- *Data Scale*: In the traditional TPC benchmarks, the testing set is frequently much larger than the actual customer data sets. Thus, the testing results can accurately indicate the real performance. However, the volume of big data is huge and ever growing; we must consider an effective way to test the production with small data sets.
- *System Evolution*: Big data growth rate is increasing; thus, big data systems must evolve accordingly to tackle the emerging requirements. Consequently, a big data benchmark must change rapidly.

B. STATUS QUO

Research on the big data benchmark remains in its infancy; these studies can be divided into two categories: component-level benchmarks and system-level benchmarks. Component-level benchmarks, also called micro-benchmarks, aim to facilitate performance comparison for a stand-alone component, whereas system-level benchmarks provide an end-to-end system testing framework. Of the components related to big data, data storage is well developed and can be modeled accurately. Thus, many micro-benchmarks have been developed for the data storage component, which can be categorized into three types:

- *TPC Benchmark*: The TPC series of benchmarks [280] have been built on the industrial consensus of representative behavior to evaluate transactional workloads for relational databases. TPCs latest decision-supporting benchmark, TCP-DS [281], covers some aspects of big data systems. Specifically, TCP-DS can generate at most 100 terabytes (current limit) of structured data, initialize the database, and execute SQL queries in both single- and multi-user modes.
- *No-SQL Benchmark*: Because unstructured data dominates the big data sets and NoSQL stores have previously demonstrated great potential in handling semi-structured and unstructured data, Yahoo! developed its cloud-serving benchmark, YCSB [159], to evaluate NoSQL stores. YCSB consists of a workload-generating client and a package of standard workloads that cover salient parts of the performance space, such as read-heavy workloads, write-heavy workloads, and scan workloads. These three workloads were run against four different data stores: Cassandra, HBase, PNUTs, and a simple shared MySQL implementation. Other research has [282], [283] extended the YCSB framework

to integrate advanced features, such as pre-splitting, bulk loading, and server-side filtering.

- *Hadoop Benchmark:* As MapReduce and its open source implementation, Hadoop, gradually become the mainstream in big data analytics, some researchers have tried to construct the TPC-like MapReduce benchmark suite with similar industrial consensus and representativeness. GridMix [284] and PigMix PigMix [285] are two built-in testing frameworks of the Apache Hadoop project, which can evaluate the performance of Hadoop clusters and Pig queries, respectively. Pavlo et al. [286] defined a benchmark consisting of a collection of tasks and compared Hadoop with two other parallel RDBMSs. The testing results reveal the performance tradeoffs and suggest that future systems should use aspects of both types of architecture. GraySort [287] is a widely used sorting benchmark that measures the performance of very large types. These benchmarks can be considered as complex superpositions of many jobs of various types and sizes. By comparing and analyzing two production MapReduce traces from Facebook and Yahoo!, Chen et al. [288] developed an open source statistical workload injector for MapReduce (SWIM). The SWIM suite includes three key components: a repository of real life MapReduce workloads, workload synthesis tools to generate representative workloads, and workload replay tools to execute the historical workloads. The SWIM suite can be used to achieve realistic workload-based performance evaluations and identify workload-specific resource bottlenecks or optimization tools. More complex analysis for production workload traces can be found in the authors' subsequent research [289].

Ghazal et al. [290] first developed an end-to-end big data benchmark, BigBench, under the product retailer model. BigBench consists of two primary components, a data generator and a query workload specification. The data generator can provide three types of raw data, structured, semi-structured, and unstructured, with scalable volumes. By borrowing the representative characteristics of the product retailer from the McKinsey report [290], the query specification defines the types of query data, data processing language, and analysis algorithms. BigBench covers the "3 Vs" characteristics of big data systems.

C. FUTURE BENCHMARK EXERCISE

The goal of testing benchmarks is to facilitate comparison of the performance of various solutions. Therefore, the development of a big data benchmark depends on mature and blooming big data systems. For a given collection of big data systems, a well-defined benchmark must choose a representative dataset as the input, model the application flow to extract the typical operations to run on the dataset, and define the evaluation metrics to compare the performance. There are two core stages, data generation and application modeling, in the evaluation procedure. In the context of big data, in addition to producing simple structured data and unstructured data, the data generator must be able to generate a high volume of data with complicated characteristics that reflect the inherent nature of UGC and social networks, including hierarchy, relevance, and rapid growth. Additionally, the application model must describe the diversity and domain correlation of big data applications, which is beyond the current abstraction, including classical queries, sorting, and data mining.

XI. CONCLUSION AND FUTURE RESEARCH

A. CONCLUSION

The era of big data is upon us, bringing with it an urgent need for advanced data acquisition, management, and analysis mechanisms. In this paper, we have presented the concept of big data and highlighted the big data value chain, which covers the entire big data lifecycle. The big data value chain consists of four phases: data generation, data acquisition, data storage, and data analysis. Moreover, from the system perspective, we have provided a literature survey on numerous approaches and mechanisms in different big data phases. In the big data generation phase, we have listed several potentially rich big data sources and discussed the data attributes. In the big data acquisition phase, typical data collection technologies were investigated, followed by big data transmission and big data pre-processing methods. In the big data storage phase, numerous cloud-based NoSQL stores were introduced, and several key features were compared to assist in big data design decisions. Because programming models are coupled with data storage approaches and play an important role in big data analytics, we have provided several pioneering and representative computation models. In the data analytics phase, we have investigated various data analytics methods organized by data characteristics. Finally, we introduced the mainstay of the big data movement, Hadoop, and big data benchmarks.

B. FUTURE RESEARCH

Many challenges in the big data system need further research attention. Below, we list the open issues covering the entire lifecycle of big data, from the big data platform and processing model to the application scenario:

- *Big Data Platform:* Although Hadoop has become a mainstay in big data analytics platforms, it remains far from mature, compared to DBMSs, which is over forty years old. First, Hadoop must integrate with real-time massive data collection & transmission and provide faster processing beyond the batch-processing paradigm. Second, Hadoop provides a concise user programming interface, while hiding the complex background execution. In some senses, this simplicity causes poor performance. We should implement a more advanced interface similar to DBMS while optimizing Hadoop performance from every angle. Third, a large-scale Hadoop cluster consists of thousands or even hundreds of thousands of servers, which means substantial energy consumption. Whether Hadoop should be widely deployed depends on its energy efficiency. Finally, privacy and security is

an important concern in the big data era. The big data platform should find a good balance between enforcing data access control and facilitating data processing.

- *Processing Model:* It is difficult for current and mature batch-processing paradigms to adapt to the rapidly growing data volume and the substantial real-time requirements. Two potential solutions are to design a new real-time processing model or a data analysis mechanism. In the traditional batch-processing paradigm, data should be stored first, and, then, the entire dataset should be scanned to produce the analysis result. Much time is obviously wasted during data transmission, storage, and repeated scanning. There are great opportunities for the new real-time processing paradigm to reduce this type of overhead cost. For instance, incremental computation attempts to analyze only the added data and combine that analysis with the original status to output the result. In-situ analysis avoids the overhead of file transfer to the centralized storage infrastructure to improve real-time performance. Due to the value-sparse feature of big data, a new data analysis mechanism can adopt dimensionality reduction or sampling-based data analysis to reduce the amount of data to be analyzed.
- *Big Data Application:* Big data research remains in its embryonic period. Research on typical big data applications can generate profit for businesses, improve the efficiency of government sectors, and promote the development of human science and technology is also required to accelerate big data progress.

REFERENCES

[1] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," in *Proc. IDC iView, IDC Anal. Future*, 2012.
[2] J. Manyika *et al.*, *Big data: The Next Frontier for Innovation, Competition, and Productivity*. San Francisco, CA, USA: McKinsey Global Institute, 2011, pp. 1–137.
[3] K. Cukier, "Data, data everywhere," *Economist*, vol. 394, no. 8671, pp. 3–16, 2010.
[4] T. economist. (2011, Nov.) *Drowning in Numbers—Digital Data Will Flood the Planet- and Help us Understand it Better* [Online]. Available: http://www.economist.com/blogs/dailychart/2011/11/bigdata-0
[5] S. Lohr. (2012). The age of big data. *New York Times* [Online]. *11*. Available: http://www.nytimes.com/2012/02/12/sunday-review/big-datas-impact-in-the-world.html?pagewanted=all&r=0
[6] Y. Noguchi. (2011, Nov.). *Following Digital Breadcrumbs to Big Data Gold*, National Public Radio, Washington, DC, USA [Online]. Available: http://www.npr.org/2011/11/29/142521910/the-digitalbreadcrumbs-that-lead-to-big-data
[7] Y. Noguchi. (2011, Nov.). *The Search for Analysts to Make Sense of Big Data*, National Public Radio, Washington, DC, USA [Online]. Available: http://www.npr.org/2011/11/30/142893065/the -search -foranalysts -to -make-%sense-of-big-data
[8] W. House. (2012, Mar.). *Fact Sheet: Big Data Across the Federal Government* [Online]. Available: http://www.whitehouse.gov/sites/default/files/microsites/ostp/big_data%_fact_sheet_3_29_2012.pdf
[9] J. Kelly. (2013). *Taming Big Data* [Online]. Available: http://wikibon.org/blog/taming-big-data/
[10] Wiki. (2013). *Applications and Organizations Using Hadoop* [Online]. Available: http://wiki.apache.org/hadoop/PoweredBy
[11] J. H. Howard *et al.*, "Scale and performance in a distributed file system," *ACM Trans. Comput. Syst.*, vol. 6, no. 1, pp. 51–81, 1988.
[12] R. Cattell, "Scalable SQL and NoSQL data stores," *SIGMOD Rec.*, vol. 39, no. 4, pp. 12–27, 2011.
[13] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.
[14] T. White, *Hadoop: The Definitive Guide*. Sebastopol, CA, USA: O'Reilly Media, 2012.
[15] J. Gantz and D. Reinsel, "Extracting value from chaos," in *Proc. IDC iView*, 2011, pp. 1–12.
[16] P. Zikopoulos and C. Eaton, *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. New York, NY, USA: McGraw-Hill, 2011.
[17] E. Meijer, "The world according to LINQ," *Commun. ACM*, vol. 54, no. 10, pp. 45–51, Aug. 2011.
[18] D. Laney, "3d data management: Controlling data volume, velocity and variety," Gartner, Stamford, CT, USA, White Paper, 2001.
[19] M. Cooper and P. Mell. (2012). *Tackling Big Data* [Online]. Available: http://csrc.nist.gov/groups/SMA/forum/documents/june2012presentations/f%csm_june2012_cooper_mell.pdf
[20] O. R. Team, *Big Data Now: Current Perspectives from O'Reilly Radar*. Sebastopol, CA, USA: O'Reilly Media, 2011.
[21] M. Grobelnik. (2012, Jul.). *Big Data Tutorial* [Online]. Available: http://videolectures.net/eswc2012_grobelnik_big_data/
[22] S. Marche, "Is Facebook making us lonely," *Atlantic*, vol. 309, no. 4, pp. 60–69, 2012.
[23] V. R. Borkar, M. J. Carey, and C. Li, "Big data platforms: What's next?" *XRDS, Crossroads, ACM Mag. Students*, vol. 19, no. 1, pp. 44–49, 2012.
[24] V. Borkar, M. J. Carey, and C. Li, "Inside big data management: Ogres, onions, or parfaits?" in *Proc. 15th Int. Conf. Extending Database Technol.*, 2012, pp. 3–14.
[25] D. Dewitt and J. Gray, "Parallel database systems: The future of high performance database systems," *Commun. ACM*, vol. 35, no. 6, pp. 85–98, 1992.
[26] (2014). *Teradata*. Teradata, Dayton, OH, USA [Online]. Available: http://www.teradata.com/
[27] (2013). *Netezza*. Netezza, Marlborough, MA, USA [Online]. Available: http://www-01.ibm.com/software/data/netezza
[28] (2013). *Aster Data*. ADATA, Beijing, China [Online]. Available: http://www.asterdata.com/
[29] (2013). *Greenplum*. Greenplum, San Mateo, CA, USA [Online]. Available: http://www.greenplum.com/
[30] (2013). *Vertica* [Online]. Available: http://www.vertica.com/
[31] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *Proc. 19th ACM Symp. Operating Syst. Principles*, 2003, pp. 29–43.
[32] T. Hey, S. Tansley, and K. Tolle, *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Cambridge, MA, USA: Microsoft Res., 2009.
[33] B. Franks, *Taming the Big Data Tidal Wave: Finding Opportunities in Huge Data Streams With Advanced Analytics*, vol. 56. New York, NY, USA: Wiley, 2012.
[34] N. Tatbul, "Streaming data integration: Challenges and opportunities," in *Proc. IEEE 26th Int. Conf. Data Eng. Workshops (ICDEW)*, Mar. 2010, pp. 155–158.
[35] (2013). *Storm* [Online]. Available: http://storm-project.net/
[36] L. Neumeyer, B. Robbins, A. Nair, and A. Kesari, "S4: Distributed stream computing platform," in *Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW)*, Dec. 2010, pp. 170–177.
[37] K. Goodhope *et al.*, "Building linkedins real-time activity data pipeline," *Data Eng.*, vol. 35, no. 2, pp. 33–45, 2012.
[38] E. B. S. D. D. Agrawal *et al.*, "Challenges and opportunities with big data—A community white paper developed by leading researchers across the united states," The Computing Research Association, CRA White Paper, Feb. 2012.
[39] D. Fisher, R. DeLine, M. Czerwinski, and S. Drucker, "Interactions with big data analytics," *Interactions*, vol. 19, no. 3, pp. 50–59, May 2012.
[40] F. Gallagher. (2013). *The Big Data Value Chain* [Online]. Available: http://fraysen.blogspot.sg/2012/06/big-data-value-chain.html
[41] M. Sevilla. (2012). *Big Data Vendors and Technologies, the list!* [Online]. Available: http://www.capgemini.com/blog/capping-it-off/2012/09/big-data-vendors-a%nd-technologies-the-list
[42] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: Distributed data-parallel programs from sequential building blocks," in *Proc. 2nd ACM SIGOPS/EuroSys Eur. Conf. Comput. Syst.*, Jun. 2007, pp. 59–72.
[43] G. Malewicz *et al.*, "Pregel: A system for large-scale graph processing," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, Jun. 2010, pp. 135–146.
[44] S. Melnik *et al.*, "Dremel: Interactive analysis of web-scale datasets," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 330–339, 2010.

[45] A. Labrinidis and H. V. Jagadish, "Challenges and opportunities with big data," *Proc. VLDB Endowment*, vol. 5, no. 12, pp. 2032–2033, Aug. 2012.
[46] S. Chaudhuri, U. Dayal, and V. Narasayya, "An overview of business intelligence technology," *Commun. ACM*, vol. 54, no. 8, pp. 88–98, 2011.
[47] (2013). *What is Big Data*, IBM, New York, NY, USA [Online]. Available: http://www-01.ibm.com/software/data/bigdata/
[48] D. Evans and R. Hutley, "The explosion of data," white paper, 2010.
[49] knowwpc. (2013). *eBay Study: How to Build Trust and Improve the Shopping Experience* [Online]. Available: http://knowwpcarey.com/article.cfm?aid=1171
[50] J. Gantz and D. Reinsel, "The digital universe decade-are you ready," in *Proc. White Paper, IDC*, 2010.
[51] J. Layton. (2013). *How Amazon Works* [Online]. Available: http://knowwpcarey.com/article.cfm?aid=1171
[52] Wikibon. (2013). *A Comprehensive List of Big Data Statistics* [Online]. Available: http://wikibon.org/blog/big-data-statistics/
[53] R. E. Bryant, "Data-intensive scalable computing for scientific applications," *Comput. Sci. Eng.*, vol. 13, no. 6, pp. 25–33, 2011.
[54] (2013). *SDSS* [Online]. Available: http://www.sdss.org/
[55] (2013). *Atlas* [Online]. Available: http://http://atlasexperiment.org/
[56] X. Wang, "Semantically-aware data discovery and placement in collaborative computing environments," Ph.D. dissertation, Dept. Comput. Sci., Taiyuan Univ. Technol., Shanxi, China, 2012.
[57] S. E. Middleton, Z. A. Sabeur, P. Löwe, M. Hammitzsch, S. Tavakoli, and S. Poslad, "Multi-disciplinary approaches to intelligently sharing large-volumes of real-time sensor data during natural disasters," *Data Sci. J.*, vol. 12, pp. WDS109–WDS113, 2013.
[58] J. K. Laurila *et al.*, "The mobile data challenge: Big data for mobile computing research," in *Proc. 10th Int. Conf. Pervas. Comput. Workshop Nokia Mobile Data Challenge, Conjunct.*, 2012, pp. 1–8.
[59] V. Chandramohan and K. Christensen, "A first look at wired sensor networks for video surveillance systems," in *Proc. 27th Annu. IEEE Conf. Local Comput. Netw. (LCN)*, Nov. 2002, pp. 728–729.
[60] L. Selavo *et al.*, "Luster: Wireless sensor network for environmental research," in *Proc. 5th Int. Conf. Embedded Netw. Sensor Syst.*, Nov. 2007, pp. 103–116.
[61] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "Sensorscope: Out-of-the-box environmental monitoring," in *Proc. IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, 2008, pp. 332–343.
[62] Y. Kim, T. Schmid, Z. M. Charbiwala, J. Friedman, and M. B. Srivastava, "Nawms: Nonintrusive autonomous water monitoring system," in *Proc. 6th ACM Conf. Embedded Netw. Sensor Syst.*, Nov. 2008, pp. 309–322.
[63] S. Kim *et al.*, "Health monitoring of civil infrastructures using wireless sensor networks," in *Proc. 6th Int. Conf. Inform. Process. Sensor Netw.*, Apr. 2007, pp. 254–263.
[64] M. Ceriotti *et al.*, "Monitoring heritage buildings with wireless sensor networks: The Torre Aquila deployment," in *Proc. Int. Conf. Inform. Process. Sensor Netw.*, Apr. 2009, pp. 277–288.
[65] G. Tolle *et al.*, "A macroscope in the redwoods," in *Proc. 3rd Int. Conf. Embedded Netw. Sensor Syst.*, Nov. 2005, pp. 51–63.
[66] F. Wang and J. Liu, "Networked wireless sensor data collection: Issues, challenges, and approaches," *IEEE Commun. Surv. Tuts.*, vol. 13, no. 4, pp. 673–687, Dec. 2011.
[67] J. Shi, J. Wan, H. Yan, and H. Suo, "A survey of cyber-physical systems," in *Proc. Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Nov. 2011, pp. 1–6.
[68] Wikipedia. (2013). *Scientific Instrument* [Online]. Available: http://en.wikipedia.org/wiki/Scientific_instrument
[69] M. H. A. Wahab, M. N. H. Mohd, H. F. Hanafi, and M. F. M. Mohsin, "Data pre-processing on web server logs for generalized association rules mining algorithm," *World Acad. Sci., Eng. Technol.*, vol. 48, p. 970, 2008.
[70] A. Nanopoulos, Y. Manolopoulos, M. Zakrzewicz, and T. Morzy, "Indexing web access-logs for pattern queries," in *Proc. 4th Int. Workshop Web Inf. Data Manag.*, 2002, pp. 63–68.
[71] K. P. Joshi, A. Joshi, and Y. Yesha, "On using a warehouse to analyze web logs," *Distrib. Parallel Databases*, vol. 13, no. 2, pp. 161–180, 2003.
[72] J. Cho and H. Garcia-Molina, "Parallel crawlers," in *Proc. 11th Int. Conf. World Wide Web*, 2002, pp. 124–135.
[73] C. Castillo, "Effective web crawling," *ACM SIGIR Forum*, vol. 39, no. 1, pp. 55–56, 2005.
[74] S. Choudhary *et al.*, "Crawling rich internet applications: The state of the art." in *Proc. Conf. Center Adv. Studies Collaborative Res. (CASCON)*, 2012, pp. 146–160.
[75] (2013, Oct. 31). *Robots* [Online]. Available: http://user-agent-string.info/list-of-ua/bots
[76] A. K. Jain, R. Bolle, and S. Pankanti, *Biometrics: Personal Identification in Networked Society*. Norwell, MA, USA: Kluwer, 1999.
[77] N. Ghani, S. Dixit, and T.-S. Wang, "On IP-over-WDM integration," *IEEE Commun. Mag.*, vol. 38, no. 3, pp. 72–84, Mar. 2000.
[78] J. Manchester, J. Anderson, B. Doshi, and S. Dravida, "Ip over SONET," *IEEE Commun. Mag.*, vol. 36, no. 5, pp. 136–142, May 1998.
[79] J. Armstrong, "OFDM for optical communications," *J. Lightw. Technol.*, vol. 27, no. 3, pp. 189–204, Feb. 1, 2009.
[80] W. Shieh, "OFDM for flexible high-speed optical networks," *J. Lightw. Technol.*, vol. 29, no. 10, pp. 1560–1577, May 15, 2011.
[81] M. Jinno, H. Takara, and B. Kozicki, "Dynamic optical mesh networks: Drivers, challenges and solutions for the future," in *Proc. 35th Eur. Conf. Opt. Commun. (ECOC)*, 2009, pp. 1–4.
[82] M. Goutelle *et al.*, "A survey of transport protocols other than standard TCP," Data Transp. Res. Group, Namur, Belgium, Tech. Rep. GFD-I.055, 2005.
[83] U. Hoelzle and L. A. Barroso, *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*, 1st ed. San Mateo, CA, USA: Morgan Kaufmann, 2009.
[84] *Cisco Data Center Interconnect Design and Deployment Guide*, Cisco, San Jose, CA, USA, 2009.
[85] A. Greenberg *et al.*, "VL2: A scalable and flexible data center network," in *Proc. ACM SIGCOMM Conf. Data Commun.*, 2009, pp. 51–62.
[86] C. Guo *et al.*, "BCube: A high performance, server-centric network architecture for modular data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 63–74, 2009.
[87] N. Farrington *et al.*, "Helios: A hybrid electrical/optical switch architecture for modular data centers," in *Proc. ACM SIGCOMM Conf.*, 2010, pp. 339–350.
[88] H. Abu-Libdeh, P. Costa, A. Rowstron, G. O'Shea, and A. Donnelly, "Symbiotic routing in future data centers," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 51–62, Oct. 2010.
[89] C. Lam, H. Liu, B. Koley, X. Zhao, V. Kamalov, and V. Gill, "Fiber optic communication technologies: What's needed for datacenter network operations," *IEEE Commun. Mag.*, vol. 48, no. 7, pp. 32–39, Jul. 2010.
[90] C. Kachris and I. Tomkos, "The rise of optical interconnects in data centre networks," in *Proc. 14th Int. Conf. Transparent Opt. Netw. (ICTON)*, Jul. 2012, pp. 1–4.
[91] G. Wang *et al.*, "c-through: Part-time optics in data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 327–338, 2010.
[92] X. Ye, Y. Yin, S. B. Yoo, P. Mejia, R. Proietti, and V. Akella, "DOS—A scalable optical switch for datacenters," in *Proc. 6th ACM/IEEE Symp. Archit. Netw. Commun. Syst.*, Oct. 2010, pp. 1–12.
[93] A. Singla, A. Singh, K. Ramachandran, L. Xu, and Y. Zhang, "Proteus: A topology malleable data center network," in *Proc. 9th ACM SIGCOMM Workshop Hot Topics Netw.*, 2010, pp. 801–806.
[94] O. Liboiron-Ladouceur, I. Cerutti, P. G. Raponi, N. Andriolli, and P. Castoldi, "Energy-efficient design of a scalable optical multiplane interconnection architecture," *IEEE J. Sel. Topics Quantum Electron.*, vol. 17, no. 2, pp. 377–383, Mar./Apr. 2011.
[95] A. K. Kodi and A. Louri, "Energy-efficient and bandwidth-reconfigurable photonic networks for high-performance computing (HPC) systems," *IEEE J. Sel. Topics Quantum Electron.*, vol. 17, no. 2, pp. 384–395, Mar./Apr. 2011.
[96] M. Alizadeh *et al.*, "Data center TCP (DCTCP)," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 63–74, 2010.
[97] B. Vamanan, J. Hasan, and T. Vijaykumar, "Deadline-aware datacenter TCP (D^2TCP)," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 115–126, 2012.
[98] E. Kohler, M. Handley, and S. Floyd, "Designing DCCP: Congestion control without reliability," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 4, pp. 27–38, 2006.
[99] H. Müller and J.-C. Freytag. (2005). Problems, methods, and challenges in comprehensive data cleansing. Professoren des Inst. Für Informatik [Online]. Available: http://www.dbis.informatik.hu-berlin.de/fileadmin/research/papers/techreports/2003-hubib164-mueller.pdf
[100] N. F. Noy, "Semantic integration: A survey of ontology-based approaches," *ACM Sigmod Rec.*, vol. 33, no. 4, pp. 65–70, 2004.
[101] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2006.

[102] M. Lenzerini, "Data integration: A theoretical perspective," in *Proc. 21st ACM SIGMOD-SIGACT-SIGART Symp. Principles Database Syst.*, 2002, pp. 233–246.
[103] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts*, vol. 4. New York, NY, USA: McGraw-Hill, 1997.
[104] M. J. Cafarella, A. Halevy, and N. Khoussainova, "Data integration for the relational web," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 1090–1101, 2009.
[105] J. I. Maletic and A. Marcus, "Data cleansing: Beyond integrity analysis," in *Proc. Conf. Inform. Qual.*, 2000, pp. 200–209.
[106] R. Kohavi, L. Mason, R. Parekh, and Z. Zheng, "Lessons and challenges from mining retail e-commerce data," *Mach. Learn.*, vol. 57, nos. 1–2, pp. 83–113, 2004.
[107] H. Chen, W.-S. Ku, H. Wang, and M.-T. Sun, "Leveraging spatio-temporal redundancy for RFID data cleansing," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2010, pp. 51–62.
[108] Z. Zhao and W. Ng, "A model-based approach for RFID data stream cleansing," in *Proc. 21st ACM Int. Conf. Inform. Knowl. Manag.*, 2012, pp. 862–871.
[109] N. Khoussainova, M. Balazinska, and D. Suciu, "Probabilistic event extraction from RFID data," in *Proc. IEEE 24th Int. Conf. Data Eng. (ICDE)*, Apr. 2008, pp. 1480–1482.
[110] K. G. Herbert and J. T. Wang, "Biological data cleaning: A case study," *Int. J. Inform. Qual.*, vol. 1, no. 1, pp. 60–82, 2007.
[111] Y. Zhang, J. Callan, and T. Minka, "Novelty and redundancy detection in adaptive filtering," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Develop. Inform. Retr.*, 2002, pp. 81–88.
[112] D. Salomon, *Data Compression*. New York, NY, USA: Springer-Verlag, 2004.
[113] F. Dufaux and T. Ebrahimi, "Video surveillance using JPEG 2000," in *Proc. SPIE*, vol. 5588. 2004, pp. 268–275.
[114] P. D. Symes, *Digital Video Compression*. New York, NY, USA: McGraw-Hill, 2004.
[115] T.-H. Tsai and C.-Y. Lin, "Exploring contextual redundancy in improving object-based video coding for video sensor networks surveillance," *IEEE Trans. Multimedia*, vol. 14, no. 3, pp. 669–682, Jun. 2012.
[116] S. Sarawagi and A. Bhamidipaty, "Interactive deduplication using active learning," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2002, pp. 269–278.
[117] Z. Huang, H. Shen, J. Liu, and X. Zhou, "Effective data co-reduction for multimedia similarity search," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2011, pp. 1021–1032.
[118] U. Kamath, J. Compton, R. I. Dogan, K. D. Jong, and A. Shehu, "An evolutionary algorithm approach for feature generation from sequence data and its application to DNA splice site prediction," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 9, no. 5, pp. 1387–1398, Sep./Oct. 2012.
[119] K. Leung *et al.*, "Data mining on DNA sequences of hepatitis B virus," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 8, no. 2, pp. 428–440, Mar./Apr. 2011.
[120] J. Bleiholder and F. Naumann, "Data fusion," *ACM Comput. Surv.*, vol. 41, no. 1, pp. 1–41, 2009.
[121] M. Günter, "Introducing MapLan to map banking survey data into a time series database," in *Proc. 15th Int. Conf. Extending Database Technol.*, 2012, pp. 528–533.
[122] K. Goda and M. Kitsuregawa, "The history of storage systems," *Proc. IEEE*, vol. 100, no. 13, pp. 1433–1440, May 2012.
[123] J. D. Strunk, "Hybrid aggregates: Combining SSDS and HDDS in a single storage pool," *ACM SIGOPS Oper. Syst. Rev.*, vol. 46, no. 3, pp. 50–56, 2012.
[124] G. Soundararajan, V. Prabhakaran, M. Balakrishnan, and T. Wobber, "Extending SSD lifetimes with disk-based write caches," in *Proc. 8th USENIX Conf. File Storage Technol.*, 2010, p. 8.
[125] J. Guerra, H. Pucha, J. S. Glider, W. Belluomini, and R. Rangaswami, "Cost effective storage using extent based dynamic tiering," in *Proc. 9th USENIX Conf. File Stroage Technol. (FAST)*, 2011, pp. 273–286.
[126] U. Troppens, R. Erkens, W. Mueller-Friedt, R. Wolafka, and N. Haustein, *Storage Networks Explained: Basics and Application of Fibre Channel SAN, NAS, ISCSI, Infiniband and FCoE*. New York, NY, USA: Wiley, 2011.
[127] P. Mell and T. Grance, "The NIST definition of cloud computing," *National Inst. Standards Technol.*, vol. 53, no. 6, p. 50, 2009.
[128] T. Clark, *Storage Virtualization: Technologies for Simplifying Data Storage and Management*. Reading, MA, USA: Addison-Wesley, 2005.
[129] M. K. McKusick and S. Quinlan, "GFS: Evolution on fast-forward," *ACM Queue*, vol. 7, no. 7, pp. 10–20, 2009.
[130] (2013). *Hadoop Distributed File System* [Online]. Available: http://hadoop.apache.org/docs/r1.0.4/hdfsdesign.html
[131] (2013). *Kosmosfs* [Online]. Available: https://code.google.com/p/kosmosfs/
[132] R. Chaiken *et al.*, "Scope: Easy and efficient parallel processing of massive data sets," *Proc. VLDB Endowment*, vol. 1, no. 2, pp. 1265–1276, 2008.
[133] D. Beaver, S. Kumar, H. C. Li, J. Sobel, and P. Vajgel, "Finding a needle in Haystack: Facebook's photo storage," in *Proc. 9th USENIX Conf. Oper. Syst. Des. Implement. (OSDI)*, 2010, pp. 1–8.
[134] (2013). *Taobao File System* [Online]. Available: http://code.taobao.org/p/tfs/src/
[135] (2013). *Fast Distributed File System* [Online]. Available: https://code.google.com/p/fastdfs/
[136] G. DeCandia *et al.*, "Dynamo: Amazon's highly available key-value store," *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 6, pp. 205–220, 2007.
[137] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web," in *Proc. 29th Annu. ACM Symp. Theory Comput.*, 1997, pp. 654–663.
[138] (2013). *Voldemort* [Online]. Available: http://www.project-voldemort.com/voldemort/
[139] (2013, Oct. 31). *Redis* [Online]. Available: http://redis.io/
[140] (2013). *Tokyo Canbinet* [Online]. Available: http://fallabs.com/tokyocabinet/
[141] (2013). *Tokyo Tyrant* [Online]. Available: http://fallabs.com/tokyotyrant/
[142] (2013, Oct. 31). *Memcached* [Online]. Available: http://memcached.org/
[143] (2013, Oct. 31). *MemcacheDB* [Online]. Available: http://memcachedb.org/
[144] (2013, Oct. 31). *Riak* [Online]. Available: http://basho.com/riak/
[145] (2013). *Scalaris* [Online]. Available: http://code.google.com/p/scalaris/
[146] F. Chang *et al.*, "Bigtable: A distributed storage system for structured data," *ACM Trans. Comput. Syst.*, vol. 26, no. 2, pp. 4:1–4:26, Jun. 2008.
[147] M. Burrows, "The chubby lock service for loosely-coupled distributed systems," in *Proc. 7th Symp. Oper. Syst. Des. Implement.*, 2006, pp. 335–350.
[148] A. Lakshman and P. Malik, "Cassandra: Structured storage system on a p2p network," in *Proc. 28th ACM Symp. Principles Distrib. Comput.*, 2009, p. 5.
[149] (2013). *HBase* [Online]. Available: http://hbase.apache.org/
[150] (2013). *Hypertable* [Online]. Available: http://hypertable.org/
[151] D. Crochford. (2006). *RFC 4627-The Application/JSON Media Type for Javascript Object Notation (JSON)* [Online]. Available: http://tools.ietf.org/html/rfc4627
[152] (2013). *MongoDB* [Online]. Available: http://www.mongodb.org/
[153] (2013). *Neo4j* [Online]. Available: http://www.neo4j.org/
[154] (2013). *Dex* [Online]. Available: http://www.sparsity-technologies.com/dex.php
[155] B. F. Cooper *et al.*, "PNUTS: Yahoo!'s hosted data serving platform," in *Proc. VLDB Endowment*, vol. 1, no. 2, pp. 1277–1288, 2008.
[156] J. Baker *et al.*, "Megastore: Providing scalable, highly available storage for interactive services," in *Proc. Conf. Innov. Database Res. (CIDR)*, 2011, pp. 223–234.
[157] J. C. Corbett *et al.*, "Spanner: Google's globally-distributed database," in *Proc. 10th Conf. Oper. Syst. Des. Implement. (OSDI)*, 2012.
[158] J. Shute *et al.*, "F1: The fault-tolerant distributed RDBMS supporting Google's ad business," in *Proc. Int. Conf. Manag. Data*, 2012, pp. 777–778.
[159] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, "Benchmarking cloud serving systems with YCSB," in *Proc. 1st ACM Symp. Cloud Comput.*, 2010, pp. 143–154.
[160] T. Kraska, M. Hentschel, G. Alonso, and D. Kossmann, "Consistency rationing in the cloud: Pay only when it matters," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 253–264, 2009.
[161] K. Keeton, C. B. Morrey, III, C. A. Soules, and A. Veitch, "Lazybase: Freshness vs. performance in information management," *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 1, pp. 15–19, Jan. 2010.
[162] D. Florescu and D. Kossmann, "Rethinking cost and performance of database systems," *ACM SIGMOD Rec.*, vol. 38, no. 1, pp. 43–48, Mar. 2009.
[163] E. A. Brewer, "Towards robust distributed systems (abstract)," in *Proc. 19th Annu. ACM Symp. Principles Distrib. Comput. (PODC)*, 2000, p. 7.

[164] S. Gilbert and N. Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services," *ACM SIGACT News*, vol. 33, no. 2, pp. 51–59, Jun. 2002.

[165] A. S. Tanenbaum and M. V. Steen, *Distributed Systems: Principles and Paradigms*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2006.

[166] L. Dagum and R. Menon, "OpenMP: An industry standard API for shared-memory programming," *IEEE Comput. Sci. & Eng.*, vol. 5, no. 1, pp. 46–55, Jan./Mar. 1998.

[167] D. W. Walker and J. J. Dongarra, "MPI: A standard message passing interface," *Supercomputer*, vol. 12, pp. 56–68, 1996.

[168] R. Pike, S. Dorward, R. Griesemer, and S. Quinlan, "Interpreting the data: Parallel analysis with sawzall," *Sci. Program.*, vol. 13, no. 4, pp. 277–298, 2005.

[169] A. F. Gates *et al.*, "Building a high-level dataflow system on top of Map-Reduce: The Pig experience," *Proc. VLDB Endowment*, vol. 2, no. 2, pp. 1414–1425, Aug. 2009.

[170] A. Thusoo *et al.*, "Hive: A warehousing solution over a Map-Reduce framework," *Proc. VLDB Endowment*, vol. 2, no. 2, pp. 1626–1629, 2009.

[171] Y. Yu *et al.*, "DryadLINQ: A system for general-purpose distributed data-parallel computing using a high-level language," in *Proc. 8th USENIX Conf. Oper. Syst. Des. Implement.*, 2008, pp. 1–14.

[172] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, and J. M. Hellerstein, "Distributed graphlab: A framework for machine learning and data mining in the cloud," *Proc. VLDB Endowment*, vol. 5, no. 8, pp. 716–727, 2012.

[173] C. Moretti, J. Bulosan, D. Thain, and P. J. Flynn, "All-pairs: An abstraction for data-intensive cloud computing," in *Proc. IEEE Int. Symp. Parallel Distrib. Process. (IPDPS)*, Apr. 2008, pp. 1–11.

[174] Y. Bu, B. Howe, M. Balazinska, and M. D. Ernst, "HaLoop: Efficient iterative data processing on large clusters," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 285–296, 2010.

[175] J. Ekanayake *et al.*, "Twister: A runtime for iterative mapreduce," in *Proc. 19th ACM Int. Symp. High Perform. Distrib. Comput.*, 2010, pp. 810–818.

[176] M. Zaharia *et al.*, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proc. 9th USENIX Conf. Netw. Syst. Des. Implement.*, 2012, p. 2.

[177] P. Bhatotia, A. Wieder, R. Rodrigues, U. A. Acar, and R. Pasquin, "Incoop: Mapreduce for incremental computations," in *Proc. 2nd ACM Symp. Cloud Comput.*, 2011, pp. 1–14.

[178] D. Peng and F. Dabek, "Large-scale incremental processing using distributed transactions and notifications," in *Proc. 9th USENIX Conf. Oper. Syst. Des. Implement.*, 2010, pp. 1–15.

[179] C. Yan, X. Yang, Z. Yu, M. Li, and X. Li, "IncMR: Incremental data processing based on mapreduce," in *Proc. IEEE 5th Int. Conf. Cloud Comput. (CLOUD)*, Jun. 2012, pp. 534–541.

[180] C. Olston *et al.*, "Nova: Continuous Pig/hadoop workflows," in *Proc. Int. Conf. Manag. Data*, 2011, pp. 1081–1090.

[181] D. G. Murray, M. Schwarzkopf, C. Smowton, S. Smith, A. Madhavapeddy, and S. Hand, "Ciel: A universal execution engine for distributed data-flow computing," in *Proc. 8th USENIX Conf. Netw. Syst. Des. Implement.*, 2011, p. 9.

[182] A. H. Eschenfelder, *Data Mining and Knowledge Discovery Handbook*, vol. 14. Berlin, Germany: Springer-Verlag, 1980.

[183] C. A. Bhatt and M. S. Kankanhalli, "Multimedia data mining: State of the art and challenges," *Multimedia Tools Appl.*, vol. 51, no. 1, pp. 35–76, 2011.

[184] G. Blackett. (2013). *Analytics Network-O.R. Analytics* [Online]. Available: http://www.theorsociety.com/Pages/SpecialInterest/AnalyticsNetwork_anal%ytics.aspx

[185] J. Richardson, K. Schlegel, B. Hostmann, and N. McMurchy. (2008). *Magic quadrant for business intelligence platforms* [Online]. Available: http://www2.microstrategy.com/download/files/whitepapers/open/Gartner-Magic-Quadrant-for-BI-Platforms-2012.pdf

[186] T. Economist. (2011). Beyond the PC, Tech. Rep. [Online]. Available: http://www.economist.com/node/21531109

[187] N. S. Foundation. (2013). *Core Techniques and Technologies for Advancing Big Data Science and Engineering* [Online]. Available: http://www.nsf.gov/pubs/2012/nsf12499/nsf12499.htm

[188] (2013). *iPlant* [Online]. Available: http://www.iplantcollaborative.org/about

[189] V. Friedman. (2008). *Data visualization and infographics* [Online]. Available: http://www.smashingmagazine.com/2008/01/14/monday-inspiration-data-visualization-and-infographics/

[190] V. Friedman. *Data visualization: Modern approaches* [Online]. Available: http://www.smashingmagazine.com/2007/08/02/data-visualization-modern-approaches/

[191] F. H. Post, G. M. Nielson, and G.-P. Bonneau, *Data Visualization: The State of the Art*. Berlin, Germany: Springer-Verlag, 2003.

[192] T. W. Anderson, T. W. Anderson, T. W. Anderson, and T. W. Anderson, *An Introduction to Multivariate Statistical Analysis*, 3rd ed. New York, NY, USA: Wiley, 2003.

[193] X. Wu *et al.*, "Top 10 algorithms in data mining," *Knowl. Inform. Syst.*, vol. 14, no. 1, pp. 1–37, 2007.

[194] G. E. Hinton, "Learning multiple layers of representation," *Trends Cognit. Sci.*, vol. 11, no. 10, pp. 428–434, 2007.

[195] G. K. Baah, A. Gray, and M. J. Harrold, "On-line anomaly detection of deployed software: A statistical machine learning approach," in *Proc. 3rd Int. Workshop Softw. Qual. Assurance*, 2006, pp. 70–77.

[196] M. Moeng and R. Melhem, "Applying statistical machine learning to multicore voltage & frequency scaling," in *Proc. 7th ACM Int. Conf. Comput. Frontiers*, 2010, pp. 277–286.

[197] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining data streams: A review," *ACM SIGMOD Rec.*, vol. 34, no. 2, pp. 18–26, Jun. 2005.

[198] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in privacy preserving data mining," *ACM SIGMOD Rec.*, vol. 33, no. 1, pp. 50–57, Mar. 2004.

[199] W. van der Aalst, "Process mining: Overview and opportunities," *ACM Trans. Manag. Inform. Syst.*, vol. 3, no. 2, pp. 7:1–7:17, Jul. 2012.

[200] G. Salton, "Automatic text processing," *Science*, vol. 168, no. 3929, pp. 335–343, 1970.

[201] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. Cambridge, MA, USA: MIT Press, 1999.

[202] A. Ritter, S. Clark, and O. Etzioni, "Named entity recognition in tweets: An experimental study," in *Proc. Conf. Empirical Methods Nat. Lang. Process.*, 2011, pp. 1524–1534.

[203] Y. Li, X. Hu, H. Lin, and Z. Yang, "A framework for semisupervised feature generation and its applications in biomedical literature mining," *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 8, no. 2, pp. 294–307, 2011.

[204] D. M. Blei, "Probabilistic topic models," *Commun. ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[205] H. Balinsky, A. Balinsky, and S. J. Simske, "Automatic text summarization and small-world networks," in *Proc. 11th ACM Symp. Document Eng.*, 2011, pp. 175–184.

[206] M. Mishra, J. Huan, S. Bleik, and M. Song, "Biomedical text categorization with concept graph representations using a controlled vocabulary," in *Proc. 11th Int. Workshop Data Mining Bioinform.*, 2012, pp. 26–32.

[207] J. Hu *et al.*, "Enhancing text clustering by leveraging wikipedia semantics," in *Proc. 31st Annu. Int. ACM SIGIR Conf. Res. Develop. Inform. Retr.*, 2008, pp. 179–186.

[208] M. T. Maybury, *New Directions in Question Answering*. Menlo Park, CA, USA: AAAI press, 2004.

[209] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Found. Trends Inform. Retr.*, vol. 2, nos. 1–2, pp. 1–135, 2008.

[210] S. K. Pal, V. Talwar, and P. Mitra, "Web mining in soft computing framework: Relevance, state of the art and future directions," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1163–1177, 2002.

[211] S. Chakrabarti, "Data mining for hypertext: A tutorial survey," *ACM SIGKDD Explorations Newslett.*, vol. 1, no. 2, pp. 1–11, 2000.

[212] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *Proc. 7th Int. Conf. World Wide Web*, 1998, pp. 107–117.

[213] D. Konopnicki and O. Shmueli, "W3QS: A query system for the world-wide web," in *Proc. 21th Int. Conf. Very Large Data Bases*, 1995, pp. 54–65.

[214] S. Chakrabarti, M. van den Berg, and B. Dom, "Focused crawling: A new approach to topic-specific web resource discovery," *Comput. Netw.*, vol. 31, nos. 11–16, pp. 1623–1640, 1999.

[215] B. Xu, J. Bu, C. Chen, and D. Cai, "An exploration of improving collaborative recommender systems via user-item subgroups," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 21–30.

[216] D. Ding *et al.*, "Beyond audio and video retrieval: Towards multimedia summarization," in *Proc. 2nd ACM Int. Conf. Multimedia Retr.*, 2012, pp. 2:1–2:8.

[217] M. Wang, B. Ni, X.-S. Hua, and T.-S. Chua, "Assistive tagging: A survey of multimedia tagging with human-computer joint exploration," *ACM Comput. Surv.*, vol. 44, no. 4, pp. 25:1–25:24, 2012.

[218] W. Hu, N. Xie, L. Li, X. Zeng, and S. Maybank, "A survey on visual content-based video indexing and retrieval," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 41, no. 6, pp. 797–819, Nov. 2011.

[219] X. Li, S. Lin, S. Yan, and D. Xu, "Discriminant locally linear embedding with high-order tensor data," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 2, pp. 342–352, Apr. 2008.

[220] X. Li and Y. Pang, "Deterministic column-based matrix decomposition," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 1, pp. 145–149, Jan. 2010.

[221] X. Li, Y. Pang, and Y. Yuan, "L1-norm-based 2DPCA," *IEEE Trans. Syst., Man, Cybern. B Cybern.*, vol. 40, no. 4, pp. 1170–1175, Apr. 2010.

[222] Y.-J. Park and K.-N. Chang, "Individual and group behavior-based customer profile model for personalized product recommendation," *Expert Syst. Appl.*, vol. 36, no. 2, pp. 1932–1939, 2009.

[223] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks," *Int. J. Approx. Reason.*, vol. 51, no. 7, pp. 785–799, 2010.

[224] Y.-G. Jiang *et al.*, "Columbia-UCF TRECvid2010 multimedia event detection: Combining multiple modalities, contextual concepts, and temporal matching," in *Proc. Nat. Inst. Standards Technol. (NIST) TRECvid Workshop*, vol. 2. 2010, p. 6.

[225] Z. Ma, Y. Yang, Y. Cai, N. Sebe, and A. G. Hauptmann, "Knowledge adaptation for ad hoc multimedia event detection with few exemplars," in *Proc. 20th Assoc. Comput. Mach. (ACM) Int. Conf. Multimedia*, 2012, pp. 469–478.

[226] J. E. Hirsch, "An index to quantify an individual's scientific research output," *Proc. Nat. Acad. Sci. United States Amer.*, vol. 102, no. 46, p. 16569, 2005.

[227] D. J. Watts, *Six Degrees: The Science of a Connected Age*. New York, NY, USA: Norton, 2004.

[228] C. C. Aggarwal, *An Introduction to Social Network Data Analytics*. Berlin, Germany: Springer-Verlag, 2011.

[229] S. Scellato, A. Noulas, and C. Mascolo, "Exploiting place features in link prediction on location-based social networks," in *Proc. 17th Assoc. Comput. Mach. (ACM) Special Interest Group Knowl. Discovery Data (SIGKDD) Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 1046–1054.

[230] A. Ninagawa and K. Eguchi, "Link prediction using probabilistic group models of network structure," in *Proc. Assoc. Comput. Mach. (ACM) Symp. Appl. Comput.*, 2010, pp. 1115–1116.

[231] D. M. Dunlavy, T. G. Kolda, and E. Acar, "Temporal link prediction using matrix and tensor factorizations," *ACM Trans. Knowl. Discovery Data*, vol. 5, no. 2, pp. 10:1–10:27, 2011.

[232] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 631–640.

[233] N. Du, B. Wu, X. Pei, B. Wang, and L. Xu, "Community detection in large-scale social networks," in *Proc. 9th WebKDD 1st SNA-KDD Workshop Web Mining Soc. Netw. Anal.*, 2007, pp. 16–25.

[234] S. Garg, T. Gupta, N. Carlsson, and A. Mahanti, "Evolution of an online social aggregation network: An empirical study," in *Proc. 9th Assoc. Comput. Mach. (ACM) SIGCOMM Conf. Internet Meas. Conf.*, 2009, pp. 315–321.

[235] M. Allamanis, S. Scellato, and C. Mascolo, "Evolution of a location-based online social network: Analysis and models," in *Proc. Assoc. Comput. Mach. (ACM) Conf. Internet Meas. Conf.*, 2012, pp. 145–158.

[236] N. Z. Gong *et al.*, "Evolution of social-attribute networks: Measurements, modeling, and implications using google+," in *Proc. Assoc. Comput. Mach. (ACM) Conf. Internet Meas. Conf.*, 2012, pp. 131–144.

[237] E. Zheleva, H. Sharara, and L. Getoor, "Co-evolution of social and affiliation networks," in *Proc. 15th Assoc. Comput. Mach. (ACM) SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 1007–1016.

[238] J. Tang, J. Sun, C. Wang, and Z. Yang, "Social influence analysis in large-scale networks," in *Proc. 15th Assoc. Comput. Mach. (ACM) SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 807–816.

[239] Y. Li, W. Chen, Y. Wang, and Z.-L. Zhang, "Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships," in *Proc. 6th Assoc. Comput. Mach. (ACM) Int. Conf. Web Search Data Mining*, 2013, pp. 657–666.

[240] T. Lappas, K. Liu, and E. Terzi, "Finding a team of experts in social networks," in *Proc. 15th Assoc. Comput. Mach. (ACM) SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 467–476.

[241] T. Zhang, A. Popescul, and B. Dom, "Linear prediction models with graph regularization for web-page categorization," in *Proc. 12th Assoc. Comput. Mach. (ACM) SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2006, pp. 821–826.

[242] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 718–729, 2009.

[243] W. Dai, Y. Chen, G.-R. Xue, Q. Yang, and Y. Yu, "Translated learning: Transfer learning across different feature spaces," in *Proc. Adv. Neural Inform. Process. Syst. (NIPS)*, 2008, pp. 353–360.

[244] M. Rabbath, P. Sandhaus, and S. Boll, "Multimedia retrieval in social networks for photo book creation," in *Proc. 1st Assoc. Comput. Mach. (ACM) Int. Conf. Multimedia Retr.*, 2011, pp. 72:1–72:2.

[245] S. Shridhar, M. Lakhanpuria, A. Charak, A. Gupta, and S. Shridhar, "Snair: A framework for personalised recommendations based on social network analysis," in *Proc. 5th Int. Workshop Location-Based Soc. Netw.*, 2012, pp. 55–61.

[246] S. Maniu and B. Cautis, "Taagle: Efficient, personalized search in collaborative tagging networks," in *Proc. Assoc. Comput. Mach. (ACM) SIGMOD Int. Conf. Manag. Data*, 2012, pp. 661–664.

[247] H. Hu, J. Huang, H. Zhao, Y. Wen, C. W. Chen, and T.-S. Chua, "Social tv analytics: A novel paradigm to transform tv watching experience," in *Proc. 5th Assoc. Comput. Mach. (ACM) Multimedia Syst. Conf.*, 2014, pp. 172–175.

[248] H. Hu, Y. Wen, H. Luan, T.-S. Chua, and X. Li, "Towards multi-screen social tv with geo-aware social sense," *IEEE Multimedia*, to be published.

[249] H. Zhang, Z. Zhang, and H. Dai, "Gossip-based information spreading in mobile networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 11, pp. 5918–5928, Nov. 2013.

[250] H. Zhang, Z. Zhang, and H. Dai, "Mobile conductance and gossip-based information spreading in mobile networks," in *IEEE Int. Symp. Inf. Theory Proc. (ISIT)*, Jul. 2013, pp. 824–828.

[251] H. Zhang, Y. Huang, Z. Zhang, and H. Dai. (2014). Mobile conductance in sparse networks and mobility-connectivity tradeoff. in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)* [Online]. Available: http://www4.ncsu.edu/?hdai/ISIT2014-HZ.pdf

[252] Cisco Syst., Inc., "Cisco visual networking index: Global mobile data traffic forecast update," Cisco Syst., Inc., San Jose, CA, USA, Cisco Tech. Rep. 2012-2017, 2013.

[253] J. Han, J.-G. Lee, H. Gonzalez, and X. Li, "Mining massive RFID, trajectory, and traffic data sets," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2008.

[254] E. Wu, Y. Diao, and S. Rizvi, "High-performance complex event processing over streams," in *Proc. Assoc. Comput. Mach. (ACM) SIGMOD Int. Conf. Manag. Data*, 2006, pp. 407–418.

[255] M. K. Garg, D.-J. Kim, D. S. Turaga, and B. Prabhakaran, "Multimodal analysis of body sensor network data streams for real-time healthcare," in *Proc. Int. Conf. Multimedia Inform. Retr.*, 2010, pp. 469–478.

[256] Y. Park and J. Ghosh, "A probabilistic imputation framework for predictive analysis using variably aggregated, multi-source healthcare data," in *Proc. 2nd Assoc. Comput. Mach. (ACM) SIGHIT Int. Health Inform. Symp.*, 2012, pp. 445–454.

[257] T. Gunarathne, T.-L. Wu, J. Qiu, and G. Fox, "Cloud computing paradigms for pleasingly parallel biomedical applications," in *Proc. 19th Assoc. Comput. Mach. (ACM) Int. Symp. High Perform. Distrib. Comput.*, 2010, pp. 460–469.

[258] Y. Bu, B. Howe, M. Balazinska, and M. D. Ernst, "Haloop: Efficient iterative data processing on large clusters," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 285–296, 2010.

[259] J. Ekanayake *et al.*, "Twister: A runtime for iterative mapreduce," in *Proc. 19th Assoc. Comput. Mach. (ACM) Int. Symp. High Perform. Distrib. Comput.*, 2010, pp. 810–818.

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

[261] T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, K. Elmeleegy, and R. Sears, "Mapreduce online," in *Proc. 7th USENIX Conf. Netw. Syst. Des. Implement.*, 2010, p. 21.

[262] B. Li, E. Mazur, Y. Diao, A. McGregor, and P. Shenoy, "A platform for scalable one-pass analytics using mapreduce," in *Proc. Assoc. Comput. Mach. (ACM) SIGMOD Int. Conf. Manag. Data*, 2011, pp. 985–996.

[263] D. Jiang, B. C. Ooi, L. Shi, and S. Wu, "The performance of mapreduce: An in-depth study," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 472–483, 2010.

[264] J. Dittrich, J.-A. Quiané-Ruiz, A. Jindal, Y. Kargin, V. Setty, and J. Schad, "Hadoop++: Making a yellow elephant run like a cheetah (without it even noticing)," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 515–529, 2010.

[265] A. Abouzied, K. Bajda-Pawlikowski, J. Huang, D. J. Abadi, and A. Silberschatz, "Hadoopdb in action: Building real world applications," in *Proc. Assoc. Comput. Mach. (ACM) SIGMOD Int. Conf. Manag. Data*, 2010, pp. 1111–1114.

[266] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling," in *Proc. 5th Eur. Conf. Comput. Syst.*, 2010, pp. 265–278.

[267] T. Sandholm and K. Lai, "Dynamic proportional share scheduling in Hadoop," in *Job Scheduling Strategies for Parallel Processing*. Berlin, Germany: Springer-Verlag, 2010, pp. 110–131.

[268] K. Kc and K. Anyanwu, "Scheduling hadoop jobs to meet deadlines," in *Proc. IEEE 2nd Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Nov./Dec. 2010, pp. 388–392.

[269] M. Yong, N. Garegrat, and S. Mohan, "Towards a resource aware scheduler in Hadoop," in *Proc. Int. Conf. Web Services (ICWS)*, 2009, pp. 102–109.

[270] S. Blanas, J. M. Patel, V. Ercegovac, J. Rao, E. J. Shekita, and Y. Tian, "A comparison of join algorithms for log processing in mapreduce," in *Proc. Assoc. Comput. Mach. (ACM) (SIGMOD) Int. Conf. Manag. Data*, 2010, pp. 975–986.

[271] J. Lin and C. Dyer, "Data-intensive text processing with mapreduce," *Synthesis Lect. Human Lang. Technol.*, vol. 3, no. 1, pp. 1–177, 2010.

[272] S. Babu, "Towards automatic optimization of mapreduce programs," in *Proc. 1st Assoc. Comput. Mach. (ACM) Symp. Cloud Comput.*, 2010, pp. 137–142.

[273] E. Jahani, M. J. Cafarella, and C. Ré, "Automatic optimization for mapreduce programs," *Proc. VLDB Endowment*, vol. 4, no. 6, pp. 385–396, 2011.

[274] J. Leverich and C. Kozyrakis, "On the energy (in) efficiency of hadoop clusters," *Assoc. Comput. Mach. (ACM) SIGOPS Operat. Syst. Rev.*, vol. 44, no. 1, pp. 61–65, 2010.

[275] W. Lang and J. M. Patel, "Energy management for mapreduce clusters," *Proc. VLDB Endowment*, vol. 3, nos. 1–2, pp. 129–139, 2010.

[276] K.-H. Lee, Y.-J. Lee, H. Choi, Y. D. Chung, and B. Moon, "Parallel data processing with mapreduce: A survey," *Assoc. Comput. Mach. (ACM) SIGMOD Rec.*, vol. 40, no. 4, pp. 11–20, 2012.

[277] B. T. Rao and L. Reddy. (2012). *Survey on improved scheduling in hadoop mapreduce in cloud environments*. arXiv preprint arXiv:1207.0780 [Online]. Available: http://arxiv.org/pdf/1207.0780.pdf

[278] (2013). *Summingbird* [Online]. Available: http://github.com/twitter/summingbird

[279] Y. Chen, "We don't know enough to make a big data benchmark suite—An academia-industry view," in *Proc. Workshop Big Data Benchmarking (WBDB)*, 2012.

[280] (2013). *TPC Benchmarks* [Online]. Available: http://www.tpc.org/information/benchmarks.asp

[281] R. O. Nambiar and M. Poess, "The making of TPC-DS," in *Proc. 32nd Int. Conf. Very Large Data Bases (VLDB) Endowment*, 2006, pp. 1049–1058.

[282] S. Patil *et al.*, "Ycsb++: Benchmarking and performance debugging advanced features in scalable table stores," in *Proc. 2nd Assoc. Comput. Mach. (ACM) Symp. Cloud Comput.*, 2011, p. 9.

[283] T. Rabl, S. Gómez-Villamor, M. Sadoghi, V. Muntés-Mulero, H.-A. Jacobsen, and S. Mankovskii, "Solving big data challenges for enterprise application performance management," *Proc. VLDB Endowment*, vol. 5, no. 12, pp. 1724–1735, 2012.

[284] (2013). *Grid Mix* [Online]. Available: http://hadoop.apache.org/docs/stable/gridmix.html

[285] (2013). *Pig Mix* [Online]. Available: http://cwiki.apache.org/PIG/pigmix.html

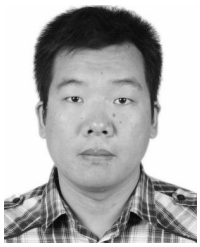
[286] A. Pavlo *et al.*, "A comparison of approaches to large-scale data analysis," in *Proc. 35th SIGMOD Int. Conf. Manag. Data*, 2009, pp. 165–178.

[287] (2013). *Gray Sort* [Online]. Available: http://sortbenchmark.org/

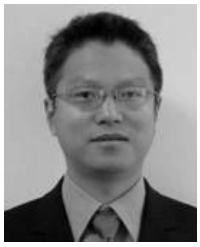
[288] Y. Chen, A. Ganapathi, R. Griffith, and R. Katz, "The case for evaluating mapreduce performance using workload suites," in *Proc. IEEE 19th Int. Symp. Model., Anal., Simul. Comput. Telecommun. Syst. (MASCOTS)*, Jul. 2011, pp. 390–399.

[289] Y. Chen, S. Alspaugh, and R. Katz, "Interactive analytical processing in big data systems: A cross-industry study of mapreduce workloads," *Proc. VLDB Endowment*, vol. 5, no. 12, pp. 1802–1813, 2012.

[290] A. Ghazal *et al.*, "Bigbench: Towards an industry standard benchmark for big data analytics," *Proc. Assoc. Comput. Mach. (ACM) SIGMOD Int. Conf. Manag. Data*, 2013, pp. 1197–1208.



HAN HU received the B.S. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2007 and 2012, respectively. He is currently a Research Fellow with the School of Computing, National University of Singapore, Singapore. His research interests include social media analysis and distribution, big data analytics, multimedia communication, and green network.



YONGGANG WEN is an Assistant Professor with the School of Computer Engineering, Nanyang Technological University, Singapore. He received the Ph.D. degree in electrical engineering and computer science (minor in Western literature) from the Massachusetts Institute of Technology, Cambridge, MA, USA. Previously, he was with Cisco Systems, Inc., San Francisco, CA, USA, to lead product development in content delivery network, which had a revenue impact of three billion U.S. dollars globally.

Dr. Wen has authored over 90 papers in top journals and prestigious conferences. His latest work in multiscreen cloud social TV has been recognized with the ASEAN ICT Award in 2013 (Gold Medal) and the IEEE Globecom Best Paper Award in 2013. He serves on editorial boards of the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE ACCESS JOURNAL, and Elsevier's *Ad Hoc Networks*. His research interests include cloud computing, green data center, big data analytics, multimedia network, and mobile computing.

TAT-SENG CHUA is currently the KITHCT Chair Professor with the School of Computing, National University of Singapore (NUS), Singapore. He was the Acting and Founding Dean of the School of Computing from 1998 to 2000. He joined NUS in 1983, and spent three years as a Research Staff Member with the Institute of Systems Science since 1980. He has worked on several multimillion-dollar projects interactive media search, local contextual search, and real-time live media search. His main research interests include multimedia information retrieval, multimedia question answering, and the analysis and structuring of user-generated contents. He has organized and served as a Program Committee Member of numerous international conferences in the areas of computer graphics, multimedia, and text processing. He was the Conference Co-Chair of ACM Multimedia in 2005, the Conference on Image and Video Retrieval in 2005, and ACM SIGIR in 2008, and the Technical PC Co-Chair of SIGIR in 2010. He serves on the editorial boards of the *ACM Transactions of Information Systems* (ACM), *Foundation and Trends in Information Retrieval* (Now), *The Visual Computer* (Springer Verlag), and *Multimedia Tools and Applications* (Kluwer). He is on the Steering Committees of the International Conference on Multimedia Retrieval, Computer Graphics International, and Multimedia Modeling Conference Series. He serves as a member of international review panels of two large-scale research projects in Europe. He is the Independent Director of two listed companies in Singapore.

XUELONG LI (M'02–SM'07–F'12) is a Full Professor with the Center for Optical Imagery Analysis and Learning, State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, China.