Setting Up a Big Data Project: Challenges, Opportunities, Technologies and Optimization

Roberto V. Zicari, Marten Rosselli, Todor Ivanov, Nikolaos Korfiatis, Karsten Tolle, Raik Niemann and Christoph Reichenbach

Abstract In the first part of this chapter we illustrate how a big data project can be set up and optimized. We explain the general value of big data analytics for the enterprise and how value can be derived by analyzing big data. We go on to introduce the characteristics of big data projects and how such projects can be set up, optimized and managed. Two exemplary real word use cases of big data projects are described at the end of the first part. To be able to choose the optimal big data tools for given requirements, the relevant technologies for handling big data are outlined in the second part of this chapter. This part includes technologies such as NoSQL and NewSQL systems, in-memory databases, analytical platforms and Hadoop based solutions. Finally, the chapter is concluded with an overview over big data

R.V. Zicari · M. Rosselli (✉) · T. Ivanov · N. Korfiatis · K. Tolle · R. Niemann · C. Reichenbach
Frankfurt Big Data Lab, Goethe University Frankfurt, Frankfurt Am Main, Germany
e-mail: rosselli@dbis.cs.uni-frankfurt.de
URL: http://www.bigdata.uni-frankfurt.de; http://www.accenture.com

R.V. Zicari
e-mail: zicari@dbis.cs.uni-frankfurt.de

T. Ivanov
e-mail: todor@dbis.cs.uni-frankfurt.de

N. Korfiatis
e-mail: n.korfiatis@uea.ac.uk

K. Tolle
e-mail: tolle@dbis.cs.uni-frankfurt.de

R. Niemann
e-mail: raik.niemann@iisys.de

C. Reichenbach
e-mail: reichenbach@em.uni-frankfurt.de

M. Rosselli
Accenture, Frankfurt, Germany

N. Korfiatis
Norwich Business School, University of East Anglia, Norwich, UK

A. Emrouznejad (ed.), *Big Data Optimization: Recent Developments and Challenges*, Studies in Big Data 18, DOI 10.1007/978-3-319-30265-2_2

benchmarks that allow for performance optimization and evaluation of big data technologies. Especially with the new big data applications, there are requirements that make the platforms more complex and more heterogeneous. The relevant benchmarks designed for big data technologies are categorized in the last part.

1 How to Set Up a Big Data Project

Data is becoming viewed as a corporate weapon [1].

How to set up and optimize a big data project?

How to set up and optimize a big data project is a challenging task that requires understanding the value of big data, the various processes and steps involved on running the project as well as the big data technologies that can be used to store and process the data. This chapter addresses these questions in three separate parts. The possible value of big data for an organization and the steps of a big data project are described in the first part. In the subsequent part the various big data technologies are classified and described according to their data model and typical use cases allowing for an easy selection of the optimal technologies given the requirements. The chapter concludes introducing big data benchmarking as a way to optimize and fine-tune the performance of the big data technologies.

Why run a big data project in the enterprise?

The most straightforward, perhaps too straightforward answer is that a big data project can offer valuable insights obtained from analyzing data, which in turn may offer a quantifiable competitive advantage to the enterprise or organization. However, beyond this high-level insight, things are less straightforward: big data projects have no single canonic use case or structure. Instead, the applications that could benefit from analyzing big data cut across industries and involve a wide variety of data sources.

As James Kobielus states [2], results from big data projects may be materialize either as revenue gains, cost reductions, or risk mitigation which have an easy and measurable Return on Investment (ROI). Often big data projects are in support of Customer Relationship Management (CRM) initiatives in marketing, customer service, sales, and brand monitoring. But of course, these are only a few examples out of many.

Big data for social good

On another approach, a different, but equally important opportunity for big data is serving the people who are in need globally, helping the society in which we live. When big data is used to improve our society and people's conditions within it, we cannot use the standard "ROI" as an impact measure but perhaps a new term such as: "SROI" or Social Return of Investments for big data?

What is big data?

Understanding and optimizing "big data" projects requires definition of what the term means. One definition that has gained considerable attention considers three distinctive characteristics of big data, namely *Volume*, *Variety* and *Velocity*, first introduced by Laney [3]. Zikopoulos and Eaton [4] summarized the dynamics and

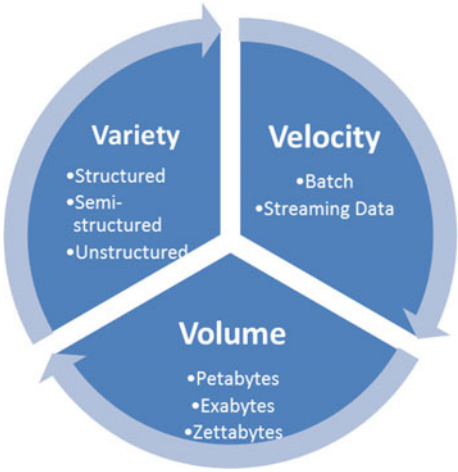

Fig. 1 IBM big data characteristics—3 V. Adopted from [4]

interconnection between these characteristics as a case of interconnected stages, known as the 3 Vs of big data. On the 3V model in Fig. 1, each stage is interconnected and runs as an input to the subsequent one.

The *Volume* represents the ever growing amount of data in petabytes, exabytes or zettabytes that is generated in today's "Internet of things" (IoT), and challenges the current state of storage systems. At the same time the *Variety* of data produced by a multitude of sources like sensors, smart devices and social media in raw, semi-structured, unstructured and rich media formats is further complicating the processing and storage of data. Finally, the *Velocity* aspect describes how quickly the data is retrieved, stored and processed.

From an information processing perspective, these three characteristics describe accurately what big data is and the new challenges that it presents to the current infrastructure. While data *Variability* and *Veracity* are also discussed as additional dimensions on this initial model [3, 4], the core Vs represent the basics for a more complete and systematic big data framework.

The emergence of *new analytical applications* and big data open new challenges [7] which will be addressed later in this chapter.

Why Big Data Analytics?

So what is so special about Big Data Analytics? Werner Vogels, CTO of Amazon.com Inc., once said in an interview [8] that "*one of the core concepts of big data is being able to evolve analytics over time*" and that "*in the new world of data analysis your questions are going to evolve and change over time and as such you need to be able to collect, store and analyze data without being constrained by resources*".

This is the fundamental difference with respect to traditional ways in Business Intelligence (BI), for example. Basically, someone discovers patterns from analyzing data, and then receives the answers to questions that in fact he/she did not ask. This sounds bizarre, but it opens up wider opportunities than simply asking the questions against structured predefined data sets, as was done before.

Jochen L. Leidner, Data Scientist at Thomson Reuters, UK explains that [9]: "*Usually analytics projects happen as an afterthought to leverage existing data created in a legacy process, which means not a lot of change of process is needed at the beginning. This situation changes once there is resulting analytics output, and then the analytics-generating process needs to be integrated with the previous processes*".

How good is the "value" that can be derived by analyzing big data?

First of all, to justify a big data project in an enterprise, in most cases there is a need to identify a *quantitative ROI.*

James Kobielus of IBM suggests [2] the use of Customer Lifetime Value (CLV) as a standard metric to evaluate, through big data analytics, the impact on customer acquisition, onboarding, retention, upsell, cross-sell and other indicators, as well as from corresponding improvements in operational efficiency. This is certainly a possibility.

Cynthia M. Saracco, also of IBM, provides the following scenario [10]: "*For example, if a telecommunications firm is able to reduce customer churn by 10 % as a result of a big data project, what's that worth? If an organization can improve the effectiveness of an email marketing campaign by 20 %, what's that worth? If an organization can respond to business requests twice as quickly, what's that worth? Many clients have these kinds of metrics in mind as they seek to quantify the value they have derived—or hope to derive—from their investment in a big data project.*"

Table 1 CRISP-DM and SEMMA process overview

CRISP-DM processes	SEMMA processes
Business Understanding: The process involves an understanding of the business problem and a consultation with domain experts on explaining what the overall goal of the project is and how it is expected to help the business function	**Sample:** The process starts with data sampling, e.g., selecting the data set for modeling. The data set should be large enough to contain sufficient information to retrieve, yet small enough to be used efficiently. This phase also deals with data partitioning
Data Understanding: This stage involves collection of initial data and exploratory analysis of the data properties such as separating data into subsets in order to form and evaluate hypotheses	**Explore:** This phase covers data understanding through the use of exploratory analytics with the help of data visualization
Data Preparation: This process concerns the preparation of data to be used for the analysis concerning the project goals	**Modify**: This stage concerns the extraction, transformation and loading (ETL) of the data to a specialized dataset
Modeling: In this stage for both models the focus is on applying various modeling (data mining) techniques on the prepared dataset in order to create models that possibly provide the desired outcome. These can, for example, be predictive or classification models targeted at predicting an output variable (e.g. forecasting sales) or classifying a composite variable (e.g. client profiling) using various features (e.g. client age, region etc.)	
Evaluation/Assessment: This process involves the evaluation of the reliability, usefulness and the impact of the models created on the business function	
Deployment: The deployment stage considers the assimilation of the project's output on the production site. This stage is not part of the SEMMA model	

How best to get started with a big data project?

Running a big data project demands an understanding of how the project differs from a typical business intelligence project that might be running concurrently within the same organization. One needs to provide an understanding of the context of the classification of such a project as "big data" using a standard definition. On the other hand, theoretical definitions of what "big data" is and how organizations and enterprises can use it has been a subject of debate [11].

The industry has identified two major methodologies for running a *data oriented project*, namely SEMMA (Sample, Explore, Modify, Model and Assess) and CRISP-DM (Cross Industry Standard Process for Data Mining). SEMMA considers in total five separate stages where in the initial stage no assistance from domain experts is required. CRISP-DM, on the other hand, considers six different stages where assistance from domain experts is expected in the initial understanding of the data. Table 1 lists the steps of the CRISP-DM and SEMMA processes respectively.

However, big data projects pose new challenges that are not covered by existing methodologies. Specifically, if we look at the business decisions that need to be made in order to successfully support a big data project in enterprise, Cynthia M. Saracco [10] and James Kobielus [2] both identify the following critical aspects:

Project's business objectives: "Begin with a clear definition of the project's business objectives and timeline, and be sure that you have appropriate executive sponsorship. The key stakeholders need to agree on a minimal set of compelling results that will impact your business; furthermore, technical leaders need to buy into the overall feasibility of the project and bring design and implementation ideas to the table."

Project scope: **"**Scope the project well to deliver near-term business benefit. Using the nucleus project as the foundation for accelerating future big data projects."

Since big data projects can get pretty complex, it is helpful to segment the work into broad categories and then drill down into each to create a solid plan.

Relevant stakeholders: "The involvement of the relevant stakeholders. It is important that the big data initiative be aligned with key stakeholder requirements. If stakeholders haven't clearly specified their requirements or expectations for your big data initiative, it's not production-ready."

Infrastructure: "In order to successfully support a big data project in the enterprise, you have to make the infrastructure and applications production-ready in your operations." In order to be able to analyze big data (i.e. data is structured and/or not structured at all) you need specific data management technology. One popular big data storage and processing technology is the Hadoop ecosystems of open software tools (see the second part of this chapter).

Skillsets: "The staff needs to have the right skillsets: e.g. database, integration and analytics skills". This point is not trivial and James Kobielus adds [2]: "Data-driven organizations succeed when all personnel—both technical and business—have a common understanding of the core big data best skills, tools and practices. You need all the skills of data management, integration, modeling, and so forth that you already have running your data marts, warehouses, OLAP cubes, and the like."

What are the problems and challenges that need to be faced in many big data projects?

The list of challenges in a big data projects include a combination of the following issues:

- Lack of appropriately scoped objectives,
- lack of required skills,
- the size of big data,
- the non-clearly defined structure of much of the big data,
- the difficulty of enforcing data consistency,
- privacy,
- data management/integration,
- rights management,
- ETL,
- data discovery (how to find high-quality data from the web?),
- data veracity (how can we cope with uncertainty, imprecision, missing details?),
- data verification,
- technical challenges for big data analytics when data is in motion rather than at rest.

Some aspects have been identified as major challenges from industry experts. Paul C. Zikopoulos of IBM comments on data velocity [12]: "*It's not just how fast data is produced or changes, but the speed at which it must be understood, acted upon, turned into something useful.*"

Scott Jarr [1] of VoltDB, a NewSQL data store company, also explains: "There is only so much static data in the world as of today. The vast majority of new data, the data that is said to explode in volume over the next 5 years, is arriving from a high velocity source. It's funny how obvious it is when you think about it. The only way to get big data in the future is to have it arrive in a high velocity rate… We think of big data velocity as data that is coming into the organization at a rate that can't be managed in the traditional database. However, companies want to extract the most value they can from that data as it arrives. We see them doing three specific things: Ingesting relentless feed(s) of data, making decisions on each piece of data as it arrives and using real-time analytics to derive immediate insights into what is happening with this velocity data."

All of these are requirements for choosing a suitable Big Data Analytical Platform.

Choosing the right data platform technology

Which technology can best scale to petabytes? Choosing the right analytics and/or data management platform for a big data project is not a trivial task. In the rest of the chapter we will outline the available data management technologies. A subsequent session on big data benchmarking will be provided as a way to measure the capacity of these technologies. Due to space limitations, we will not cover the analytics aspect in detail.

For a Data Management Platform, the key requirement is the ability to scale. Scalability has three aspects: data volume, hardware size and concurrency.

It is important to note that scale and performance requirements for big data strain conventional relational databases. As a result, new generations of NoSQL database systems, new implementation of relational databases (also called NewSQL), in-memory databases, and graph databases have emerged. In addition to this, there is of course the entire eco-system of open source software related to Hadoop.

Why Hadoop?

John Schroeder, CEO and co-founder of MapR Technologies, explains [13]: "One of the benefits of Hadoop is that you don't need to understand the questions you are going to ask ahead of time, you can combine many different data types and determine required analysis you need after the data is in place."

Daniel Abadi, Co-founder of Hadapt, further explains [14]: "A lot of people are using Hadoop as a sort of data refinery. Data starts off unstructured, and Hadoop jobs are run to clean, transform, and structure the data. Once the data is structured, it is shipped to SQL databases where it can be subsequently analyzed. This leads to the raw data being left in Hadoop and the refined data in the SQL databases."

> But it's basically the same data—one is just a cleaned (and potentially aggregated) version of the other. Having multiple copies of the data can lead to all kinds of problems. For example, let's say you want to update the data in one of the two locations—it does not get automatically propagated to the copy in the other silo. Furthermore, let's say you are doing some analysis in the SQL database and you see something interesting and want to drill down to the raw data—if the raw data is located on a different system, such a drill down becomes highly nontrivial. Furthermore, data provenance is a total nightmare. It's just a really ugly architecture to have these two systems with a connector between them.

Hybrid architectures

This result is, as James Kobielus comments [2]: "In the larger evolutionary perspective, big data is evolving into a hybridized paradigm under which Hadoop, massively parallel processing (MPP) enterprise data warehouses (EDW), in-memory columnar, stream computing, NoSQL, document databases, and other approaches support extreme analytics in the cloud. Hybrid architectures address the heterogeneous reality of big data environments and respond to the need to incorporate both established and new analytic database approaches into a common architecture. The fundamental principle of hybrid architectures is that each constituent big data platform is fit-for-purpose to the role for which it's best suited."

The "big data platform" that is to be used needs to meet the availability, security and robustness requirements expected of the enterprise infrastructure. This affects the entire big data technology stack: databases, middleware, applications, and tools. The environment has to be designed for modular scaling in order to cope with the growth in data volumes, velocities and varieties.

Is Open Source the answer?

A lot of the software for handling big data is open source, but of course not only. Cynthia M. Saracco of IBM mentions that [10]: "Even if your big data solution uses open source software, there are still expenses involved for designing, developing, deploying, and maintaining your solution. So what did your business gain from that investment? The answer to that question is going to be specific to your application and your business".

Elastic computing in the cloud? How does it relate to Big Data Analytics?

Cloud computing and virtualization of resources are directly related to big data. Paul C. Zikopoulos [12] comments: "This is pretty important because I need the utility-like nature of a Hadoop cluster, without the capital investment. Time to analytics is the benefit here.

"After all, if you're a start-up analytics firm seeking venture capital funding, do you really walk into to your investor and ask for millions to set up a cluster; you'll get kicked out the door. No, you go to Rackspace or Amazon, swipe a card, and get going. IBM is there with its Hadoop clusters (private and public) and you're looking at clusters that cost as low as $0.60 US an hour.

> I think at one time I costed out a 100 node Hadoop cluster for an hour and it was like $34US—and the price has likely gone down. What's more, your cluster will be up and running in 30 min.

Use cases from big data projects

To end the first part of this chapter, we will illustrate two use cases as examples of how big data is used to create value for a company:

Use Case 1: NBCUniversal International

Dr. Bassett, Director of Data Science for NBCUniversal International, UK, explains the process by which they "dig into" data [15]:

> I'm the Director of Data Science for NBCUniversal International. I lead a small but highly effective predictive analytics team. I'm also a "data evangelist"; I spend quite a bit of my time helping other business units realize they can find business value from sharing and analyzing their data sources.
>
> We predict key metrics for the different businesses—everything from television ratings, to how an audience will respond to marketing campaigns, to the value of a particular opening weekend for the box office. To do this, we use machine learning regression and classification algorithms, semantic analysis, monte-carlo methods, and simulations.
>
> We start with the insight in mind: What blind-spots do our businesses have, what questions are they trying to answer and how should that answer be presented? Our process begins with the key business leaders and figuring out what problems they have—often when they don't yet know there's a problem. Then we start our feature selection, and identify which sources of data will help achieve our end goal—sometimes a different business unit has it sitting in a silo and we need to convince them to share, sometimes we have to build a system to crawl the web to find and collect it.
>
> Once we have some idea of what we want, we start brainstorming about the right methods and algorithms we should use to reveal useful information: Should we cluster across a multi-variate time series of market response per demographic and use that as an input for a regression model? Can we reliably get a quantitative measure of a demographics engagement from sentiment analysis on comments? This is an iterative process, and we spend quite a bit of time in the "capturing data/transforming the data" step.
>
> We predict key metrics for the different businesses—everything from television ratings, to how an audience will respond to marketing campaigns, to the value of a particular opening weekend for the box office. To do this, we use machine learning regression and classification algorithms, semantic analysis, monte-carlo methods, and simulations. For instance, our cinema distribution company operates in dozens of countries. For each day in each one, we need to know how much money was spent and by whom -and feed this information into our machine-learning simulations for future predictions. Each country might have dozens more cinema operators, all sending data in different formats and at different qualities. One

territory may neglect demographics, another might mis-report gross revenue. In order for us to use it, we have to find missing or incorrect data and set the appropriate flags in our models and reports for later. Automating this process is the bulk of our big data operation. Big data helps everything from marketing, to distribution, to planning. In marketing, we know we're wasting half our money. The problem is that we don't know which half. Big data is helping us solve that age-old marketing problem. We're able to track how the market is responding to our advertising campaigns over time, and compare it to past campaigns and products, and use that information to more precisely reach our audience (a bit how the Obama campaign was able to use big data to optimize its strategy). In cinema alone, the opening weekend of a film can affect gross revenue by seven figures (or more), so any insight we can provide into the most optimal time can directly generate thousands or millions of dollars in revenue.

Being able to distill big data from historical information, audiences responses in social media, data from commercial operators, et cetera, into a useable and interactive simulation completely changes how we plan our strategy for the next 6–15 months.

Use Case 2: Thomson Reuters

Dr. Jochen L. Leidner, Lead Scientist of the London R&D at Thomson Reuters, UK explains [9]:

For the most part, I carry out applied research in information access, and that's what I have been doing for quite a while. I am currently a Lead Scientist with Thomson Reuters, where I am building up a newly-established London site part of our Corporate Research & Development group.

Let me say a few words about Thomson Reuters before I go more into my own activities, just for background. Thomson Reuters has around 50,000 employees in over 100 countries and sells information to professionals in many verticals, including finance & risk, legal, intellectual property & scientific, tax & accounting. Our headquarters are located at 3 Time Square in the city of New York, NY, USA. Most people know our REUTERS brand from reading their newspapers (thanks to our highly regarded 3,000 + journalists at news desks in about 100 countries, often putting their lives at risk to give us reliable reports of the world's events) or receiving share price information on the radio or TV, but as a company, we are also involved in as diverse areas as weather prediction (as the weather influences commodity prices) and determining citation impact of academic journals (which helps publishers sell their academic journals to librarians), or predicting Nobel prize winners.

My research colleagues and I study information access and especially means to improve it, using including natural language processing, information extraction, machine learning, search engine ranking, recommendation system and similar areas of investigations. We carry out a lot of contract research for internal business units (especially if external vendors do not offer what we need, or if we believe we can build something internally that is lower cost and/or better suited to our needs), feasibility studies to de-risk potential future products that are considered, and also more strategic, blue-sky research that anticipates future needs. As you would expect, we protect our findings and publish them in the usual scientific venues.

Thomson Reuters is selling information services, often under a subscription model, and for that it is important to have metrics available that indicate usage, in order to inform our strategy. So another example for data analytics is that we study how document usage can inform personalization and ranking, of from where documents are accessed, and we use this to plan network bandwidth and to determine caching server locations.

For most definitions of "big", yes we do have big data. Consider that we operate a news organization, which daily generates in the tens of thousands of news reports (if we count all languages together). Then we have photo journalists who create large numbers of high-quality, professional photographs to document current events visually, and videos

comprising audio-visual storytelling and interviews. We further collect all major laws, statutes, regulations and legal cases around in major jurisdictions around the world, enrich the data with our own meta-data using both manual expertise and automatic classification and tagging tools to enhance findability. We hold collections of scientific articles and patents in full text and abstracts. We gather, consolidate and distribute price information for financial instruments from hundreds of exchanges around the world. We sell real-time live feeds as well as access to decades of these time series for the purpose of back-testing trading strategies.

Big data analytics lead to cost savings as well as generate new revenues in a very real, monetary sense of the word "value". Because our solutions provide what we call "knowledge to act" to our customers, i.e., information that lets them make better decisions, we provide them with value as well: we literally help our customers save the cost of making a wrong decision.

Even with new projects, product managers still don't think of analytics as the first thing to build into a product for a first bare-bones version, and we need to change that; instrumentation is key for data gathering so that analytics functionality can build on it later on. In general, analytics projects follow a process of (1) capturing data, (2) aligning data from different sources (e.g., resolving when two objects are the same), (3) pre-processing or transforming the data into a form suitable for analysis, (4) building some model and (5) understanding the output (e.g. visualizing and sharing the results). This five-step process is followed by an integration phase into the production process to make the analytics repeatable.

Where are we now?

The IBM Institute for Business Value did a joint study [16] with Said Business School (University of Oxford) on the adoption of big data technologies in enterprise. They found that 28 % were in the pilot phase, 24 % haven't started anything, and 47 % are planning.

The rest of this chapter is organized as follows. Section (2) provides an overview of big data storage technologies. Section (3) provides an overview of big data benchmarking and its use in assessing different platforms. Section (4) provides conclusions and key points with respect to the contributions of this chapter.

2 Big Data Management Technologies

Given the high volume, velocity and variety of big data, the traditional Data Warehouse (DWH) and Business Intelligence (BI) architectures already existing in companies need to be enhanced in order to meet the new requirements of storing and processing big data. To optimize the performance of the Big Data Analytics pipeline, it is important to select the appropriate big data technology for given requirements. This section contains an overview of the various big data technologies and gives recommendations when to use them.

A survey by Forrester Research [17] indicated that most companies are relying on a mix of different technologies to enable the storage and processing of big data. Besides traditional relational data warehouse and business intelligence technologies that already exist in most companies, a big data architecture often includes non-relational technologies, new-relational technologies, in-memory databases,

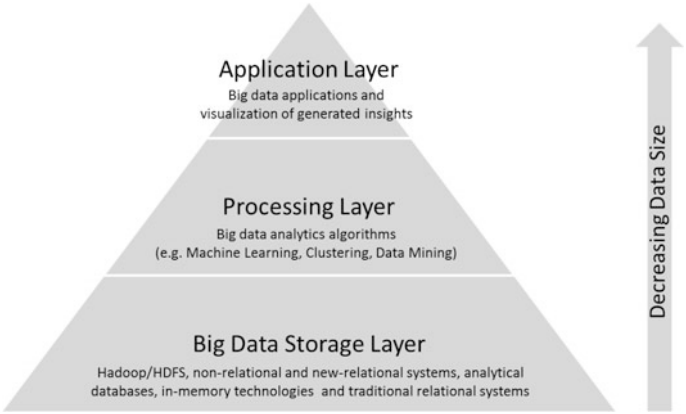

Fig. 2 The big data stack divided into three different layers

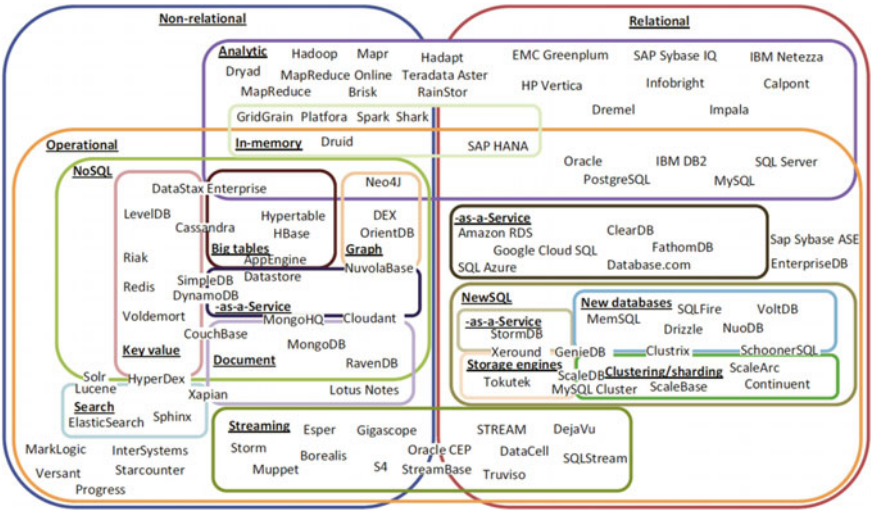

Fig. 3 Landscape and categorization of the high variety of existing database systems [18]

analytical platforms, Hadoop based solutions as well as big data streaming technologies.

In this section we focus on the technologies for storing and handling big data (the big data storage layer in Fig. 2) that analytical algorithms will subsequently use to generate meaningful insights.

There are a lot of different technologies and database systems to choose from when setting up a big data infrastructure. Figure 3 depicts a rough categorization of the high variety of different existing systems. The biggest categories and the core technologies for big data are covered in this chapter.

Particularly, we will outline the following types of technologies:

- Non-relational technologies (NoSQL)
- New-relational technologies (NewSQL)
- In-memory Databases
- Analytical platforms
- Hadoop based solutions
- Streaming technologies

2.1 NoSQL Systems

According to Cattell [19], the term NoSQL ("Not only SQL") describes a new type of data stores that are specifically designed to meet the requirements that come with applications usually running in a distributed environment with many thousands of concurrent users, e.g. Web 2.0 applications.

Cattell [19] states that these NoSQL systems are designed to deal with updating and reading heavy OLTP workloads, and generally have six distinct properties:

- Ability to scale horizontally
- Distribution and replication of data over many servers
- Simple interfaces, not necessary SQL
- Weaker concurrency models than ACID
- Utilization of distributed indexes and memory
- Flexible schemata

As stated in the so called CAP theorem [20], it is not possible for a distributed system to archive the three properties of consistency (C), availability (A) and partition tolerance (P) at the same time. It is only possible to fully achieve two of those properties at once. Hence in practice a tradeoff between the properties has to be made.

As aforementioned, NoSQL systems usually run in a distributed environment and thus need to be partition tolerant. At the same time they need to be highly available for thousands of users. Those two properties can fully be achieved only by using a weaker consistency model. For this reason, most NoSQL systems do not support ACID transactions [21], but instead use a weaker consistency model such as eventual consistency [22, 23], also called BASE (Basically Available, Soft State, Eventual Consistency). This consistency model guarantees that if no new updates are made on a data record, all reads on this record will eventually return the last updated value [24]. The BASE consistency model is used in most modern NoSQL systems. Nevertheless, there are systems that still use an ACID consistency model, but at the price of a lower availability or partition tolerance.

The various NoSQL systems are widely categorized into four groups according to their data model: *key-value stores, document-oriented stores, wide column stores and graph databases* [25]. An overview is provided below:

Key	Value
1	John, Doe, New York, 44, 11-05-1970
2	Frank, Doe, San Fransicso, 49, 08-06-1965
3	Dave, Doe, Melbourne, 38, 02-12-1976

Fig. 4 Key-value data model

Key-value stores: Key-value stores come along with a very simple data model. Each record consists of a key and a value (Fig. 4). The key and the value can be any value depending on the implementation of a specific database, for example a string or an integer value.

Famous examples of key-value stores are Cassandra, Riak, Amazon Dynamo, Voldemort, Redis and BarkelyDB [25].

Cassandra is one of the most prominent key-value stores and is used by companies like Digg and Reddit. This system is able to process big data workloads across multiple nodes with no single point of failure. Cassandra addresses the problem of failures by employing a peer-to-peer distributed system where all nodes are the same and data is distributed among all nodes in the cluster. Each node exchanges information across the cluster every second. A commit log on each node captures write activity to ensure data durability [26]. This high level of distribution is possible due to the use of a simple data model. Cassandra uses a very unique tunable consistency model with different consistency levels where the database administrator can set the number of replica and favor consistency over availability, or vice versa. In practice this results in a tradeoff between data accuracy and response times. Cassandra has a flexible schema and comes with its own query language called Cassandra Query Language (CQL) [27].

The simple key-value data model allows for a very fast read/write performance and good scalability over various servers. Key-value stores are a good choice for very fast and highly scalable retrieval of values, and when a high availability is needed, for example for tasks such as managing user profiles or retrieving product names. Amazon have developed their own key-value store Dynamo for their shopping cart for these reasons [25].

Document-oriented databases: As the name implies, a document-oriented database or document store deals with documents as its smallest aggregation unit. A document in such a data store can have for example a JSON, BSON or XML format depending on the database implementation. Every document consists of various attributes without a fixed schema. Figure 5 depicts how the structure of a document could look like.

The open source database MongoDB is a popular example of a document store. Documents in MongoDB have a JSON format and are stored physically in a binary version (BSON). Indexes are supported for all attributes in any document created in the database and documents can be loaded into the database without first defining a schema. The data is automatically replicated over different nodes thus giving it a high availability. The database follows an eventual consistency data model with

Fig. 5 Document structure in JSON

```
{
      "id": „6f38gd59hv48gj95s",
      "FirstName": "John",
      "LastName": "Doe",
      "City": "New York",
      "Age": "44"
      "Birthday": "10-05-1970",
}
```

MongoDB providing different interfaces such as JavaScript, HTTP or REST [28]. Similar well-known document stores are CouchDB, MarkLogic and RavenDB.

Document stores in general are useful for storing large collections of documents or semi-structured data like text documents, XML/JSON documents or email messages. The documents stored in the database do not have to share the same structure and thus empty values can be avoided in contrast to a RDBMS [25].

Wide column stores: In a wide column or column family data store, the data is stored in a gigantic sparse table that can easily be split and distributed over various servers. Such a table can have millions of columns containing many empty entries for various columns of a data record. The data is saved in a column-oriented fashion with multiple attributes per key. The most prominent examples of this type of data model are BigTable from Google and its open source equivalent Apache HBase [25].

Figure 6 depicts the data structure of HBase. Each row can be addressed by its unique row key. The columns are grouped in column families where the first column ("data") forms its own column family "data", while the last two columns ("mimetype" and "size") form the column family "meta". Additionally each data record is time-versioned and has a timestamp. Originally this data structure was used by Google in their BigTable implementation to store all website information from their crawling operations (including time-versioned data and the history of a website) needed for their search engine [29].

The data structure shown in Fig. 6 is physically saved in a sparse table in a column-oriented fashion. The column-oriented way of storing the data avoids large

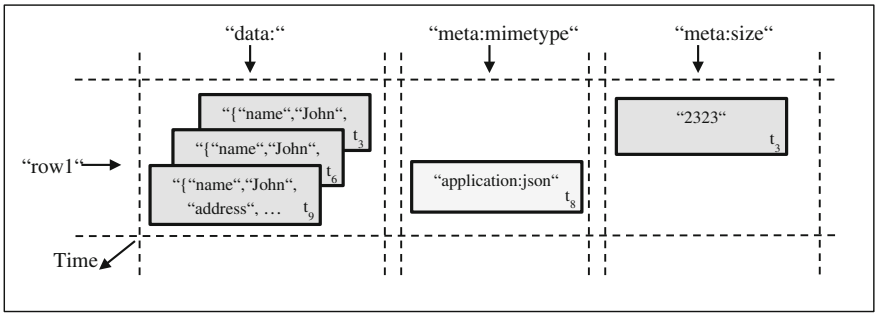

Fig. 6 Apache HBase data structure. Adopted from [30]

numbers of empty entries (null values), which would occur for all the missing timestamp entries, or any missing entries in general. HBase uses a strong consistency model avoiding the problems relating to an eventual consistency model. The data is distributed automatically over different servers with linear scalability and including automatic failover. Being based on HDFS and being part of the Hadoop ecosystem from the beginning, Hadoop/HDFS and MapReduce are supported natively. HBase can be accessed programmatically with Java or via a REST API [31].

Wide column stores are ideal for distributed data storage with or without versioned data, as well as large scale and batch-oriented data processing such as sorting, parsing and transformation of data. They are also suitable for purposes of exploratory and predictive analytics [25].

Graph databases: Even for a NoSQL database, graph databases have a very unique data model and are special in terms of scalability and their use cases. A graph consists of nodes connected by edges forming a network. Graph databases are useful for very complex queries traversing the network for which you would otherwise need to join multiple tables in a traditional relational database. Examples of graph databases are AllegroGraph, Neo4 J, DEX, Sones and HyperGraphDB [32].

To be precise, it is necessary to state that the data model behind the different graph databases might differ from system to system, depending on the supported graph structure. This ranges from allowing attributions to nodes or edges, to the support of hypergraphs (where an edge not only can connect one node with another but can interconnect groups of nodes with each other) [33].

For various applications a graph is the native underlying data structure. In the context of big data, one automatically thinks of social networks where persons are represented by nodes connected to their friends, or the structure of the Web where the links between them are the edges (used for example to calculate *page rank*) [34]. The more someone starts thinking about graphs and their expressive power, the more he/she realizes that there are many more applications where graphs can be used. In fact, the object-oriented world with objects as nodes and their associations building edges suggests that graphs could be used for nearly any modern application. This provides there is a very high potential, nevertheless merging to a graph database involves some effort and it depends on the queries you plan to run on your system whether this pays off or not.

On the level of querying graph databases, there are a number of existing query languages available, including SPARQL [35], Cypher, Gremlin [36].

2.2 NewSQL Systems

NewSQL systems form another category of modern database systems. Databases in this category support the ACID features known from traditional relational databases, but unlike them they are designed with the goal of running in distributed environments with a good horizontal scalability.

VoltDB is a typical example of a NewSQL system. In VoltDB the data and the processing associated with it are distributed over the nodes in a cluster. Each node holds a subset of the data and the corresponding stored procedures to process the data. This makes VoltDB a good choice for OLTP style workloads, which require good scalability, high availability and high throughput. However, VoltDB is not optimized for OLAP workloads and analytical queries where a lot of data has to be called up from the database. This system is thus not a good choice as a database for business intelligence or similar applications [37]. Note that VoltDB also contains in-memory technologies and can thus be labeled as an in-memory database (see the next section) as well.

Other prominent examples of NewSQL databases are FoundationDB and NuoDB.

2.3 In-Memory Databases

The extreme performance potentials of in-memory database management system technologies are very attractive to organizations when it comes to real-time or near real-time processing of large amounts of data. In a report by Gartner [38], in-memory infrastructures are defined as follows: "*In-memory-enabling application infrastructure technologies consist of in-memory database management systems, in-memory data grids, high-performance messaging infrastructures, complex-event processing platforms, in-memory analytics and in-memory application servers. These technologies are being used to address a wide variety of application scenarios requiring a combination of speed of execution, scalability and insightful analytics.*"

A good example of an in-memory database is SAP HANA. All data in HANA resides in the memory, which allows for faster query execution compared to traditional disk-based technologies. In HANA disks are only used for backup and recovery purposes.

Other famous examples of in-memory databases are VoltDB and Aerospike.

2.4 Analytical Platforms

Analytical platforms are solutions specifically designed to meet the requirements of advanced analytics and OLAP workloads for huge amounts of data. Gartner defines advanced analytics as "the analysis of all kinds of data using sophisticated quantitative methods (for example, statistics, descriptive and predictive data mining, simulation and optimization) to produce insights that traditional approaches to Business Intelligence (BI) - such as query and reporting - are unlikely to discover" [39]. Gartner further states that"predictive analytics and other categories of advanced analytics are becoming a major factor in the analytics market" [39].

Commercial analytical platforms, which are also called appliances, usually consist of a predefined combination of hardware and software components that can be used in an out of the box deployment. Equivalent open source implementations mostly consist only of software components that can run on any hardware but need further configuration. Examples of analytical platforms are Teradata Aster, IBM Netezza or HP Vertica.

2.5 Hadoop Based Solutions

The core components of Apache Hadoop are the Hadoop Distributed File System (HDFS) [40], inspired by the Google File System (GFS) [41], and the MapReduce programming framework, based on Googles MapReduce algorithm [42]. Additional Hadoop components such as HBase [30] (a wide column store on top of HDFS), Hive [43] (support for SQL queries) and Pig [44] (support for writing MapReduce programs) have been developed on top of Hadoop and make up the so-called Hadoop ecosystem.

A study by Yahoo [45] describes the architecture of Hadoop as follows: "Every HDFS cluster consists of a single master node (NameNode) and multiple, up to thousands, DataNodes who store the data."

The data is stored in files divided in large blocks (typically 128 MB) which are replicated over multiple DataNodes. The replication factor (usually three) is adjustable and can be specified by the user. User interaction with the file system is done using a HDFS code library. Via this library a user can read, write or delete files and directories within the Hadoop file system without being concerned with the different data locations within the cluster [45].

The MapReduce framework allows for parallel processing of the data in HDFS. The processing of the data is broken down into the map and the reduce phases, which in turn allows parallelization. In the map phase the input data is distributed over the map processes, which are also called tasks. A single map task can process its part of the data independently of the other. The purpose of the reduce tasks is to combine the results from all the map tasks and calculate the overall result [46].

In Hadoop it is possible to store and process petabytes of unstructured data in a batch mode. Since the data is stored in the HDFS blocks, no schema definition is required and new nodes can be added at any time for linear increase of the storage space. This makes Hadoop a good choice for a distributed data hub, for example. When the data arrives it can be stored in the HDFS as a first landing zone, then it can be refined by MapReduce jobs in order to eventually transfer parts of the refined data to the traditional Data Warehousing (DWH) systems for analysis with existing BI tools. Recent developments like Apache Spark [47] enhance Hadoop to allow not only pure batch processing, but also interactive data exploration [48].

2.6 Big Data Streaming Systems

Many big data tasks require access not only to static, rarely-changing data, but also to continuous streams of data that represent live events, such as RSS feeds, microblogging feeds, sensor data, or other bus or network events. When off-line batch processing is not an option, systems with explicit support for stream processing can provide numerous benefits:

- reduced computational overhead [49],
- automatic resource re-use through shared processing [50],
- decreased latency, permitting near-real-time notification delivery to human observers (e.g. via dashboards) or automated actors, and
- query language support for streaming.

The principal differentiating factor and challenge for such streaming big data systems is time: new input tuples are continuously coming in through a family of streams, placing different requirements on semantics and evaluation strategies to those for off-line databases. On the other hand, output may also never finish, so that the output of a streaming big data system is another family of streams. The query processor provides plumbing to connect these streams with operators and functions, and may provide facilities to add and remove computations in flight [51], i.e. without disturbing ongoing processing.

As streams may continue to be processed without interruption for arbitrary lengths of time, stream processing cannot retain a full history of all past input tuples. This limits the available operations; while stream-based systems can easily support, for example, UNION operators or other stream merges, arbitrary joins over the entire history are not necessarily feasible. However, many important queries over input streams do not require full historical knowledge, only knowledge of data from the last hour or day. Contemporary querying approaches exploit this insight through *sliding windows* [50, 52, 53] into data streams, which capture all data from a given stream within a specified time frame relative to the current point in time. As time progresses, the window slides forward and the computation is updated to reflect the new contents of the window. Analogously, aggregation can be interpreted as a rolling aggregate, reflecting the status of aggregation at a given point in time.

Such stream processing requires continuous computations. If these begin to overwhelm an individual node, the management process may launch additional processing nodes to increase processing bandwidth. As with off-line queries, such parallel processing can be realized effectively for associative operators. Furthermore, the system may overlap the computation of multiple sliding windows into the same data stream, but at different timestamps, scheduling the windows on different nodes [54]. Alternatively, some application domains may be amenable to dropping input tuples that are not strictly required for the system to function correctly, deliberately degrading output quality according to Quality-of-Service rules [55].

Several streaming big data systems are available today, among them Open Source solutions such as Apache Spark [56] and Storm [57]. Commercial

service-based solutions include Amazon Kinesis [58], which is designed to run as part of the Amazon Web Services framework and Google BigQuery. Forrester Research [59] lists several additional commercial platforms, including offerings from IBM, SAP, Software AG, and Tibco [60].

3 Big Data Benchmarking

In this section we focus on the question of how big data technologies can be optimized and evaluated in terms of the performance requirements of big data applications.

3.1 Why Do We Need Big Data Benchmarking?

Choosing the right big data platform and configuring it properly to provide the best performance for a hosted application is not a trivial task.

Especially with the new big data applications, there are requirements that make the platforms more complex, more heterogeneous, and hard to monitor and maintain. The role of benchmarking becomes even more relevant as a method for evaluating and understanding better the internals of a particular platform. Furthermore, benchmarks are used to compare different systems using both technical and economic metrics that can guide the user in the process of finding the right platform that fits their needs.

Nevertheless, the user has to first identify his needs and then choose the ideal big data benchmark. Big Data Benchmarks are a good way to optimize and fine-tune the performance in terms of processing speed, execution time or throughput of the big data system. A benchmark can also be used to evaluate the availability and fault-tolerance of a big data system. Especially for distributed big data systems a high availability is an important requirement.

While some benchmarks are developed to test particular software platforms, others are technology independent and can be implemented for multiple platforms. Usually the technology specific benchmarks are used to simulate specific types of applications, which will be hosted on the platform and should run in an optimal way.

At the Frankfurt Big Data Lab we use benchmarks to not only evaluate the performance of big data platforms [61, 62], but also to evaluate the availability and fault-tolerance [63].

3.2 Big Data Benchmarking Challenges

For many years the benchmarks specified by the Transaction Processing Performance Council (TPC) [64] have been successfully used as a standard for comparing

OLTP and OLAP systems. Just recently the TPC have formed a new group for the standardization of a big data benchmark [65] along with other similar initiatives like the Big Data Top 100 [66] and the Big Data Benchmarking Community [67]. However, the existing and emerging big data applications and platforms have very different characteristics ("3Vs") compared to the traditional transactional and analytical systems. These new platforms can store various types of data (structured, unstructured or semi-structured) with the schema (schema-on-read) defined just before accessing the data. They also support different types of data processing: batch, real-time or near real-time. The large data volumes force its distribution among multiple nodes and the use of additional fault tolerance techniques to guarantee data reliability. At the same time, many new applications that deal with the data are employed, leading to increased workload diversity on the big data systems.

All these big data challenges can make the systems very complex and difficult to standardize, which is a major objective when defining a benchmark. Furthermore it is still not clear if the benchmark should target single system components by using micro-benchmarks, or, on the contrary, it should include an end-to-end benchmark suite which stress tests the entire platform with all types of workloads. Moreover, the metrics provided by big data benchmarks should be extensive and comparable among the multiple systems under test. Chen et al. [68] outline four unique challenges of systems that hinder the development of big data benchmarks: (i) *system complexity*; (ii) *use case diversity*; (iii) *data scale*, which makes reproducing behavior challenging; and (iv) *rapid system evolution*, which requires the benchmark to keep pace with changes in the underlying system. Similarly, Xiong et al. [69] identify three key considerations that a big data benchmark should meet: (i) a benchmark suite should have workloads that are *representative of a wide range of application domains*; (ii) workloads in a benchmark suite should have *diversity of data characteristics*; and (iii) a benchmark suite *should not have redundant workloads in itself.*

3.3 Big Data Benchmarking Comparison

There are numerous projects that identify and compare the main components of a big data benchmark. In their paper presenting the CloudRank-D benchmark, Luo et al. [70] consider a number of workload characteristics which a benchmark suite should meet. These characteristics are listed in Table 2 and compared with similar benchmark suites.

However, as the CloudRank-D covers various representative applications it cannot really be compared with function-specific benchmarks like WL suite (also called SWIM), which analyzes the workload characteristics based on the number of jobs, arrival pattern and computation using synthesized representative data from real MapReduce traces.

Other important arguments discussed in the paper are the accuracy of the reported benchmark metrics and the target platforms that it can evaluate. In Table 3,

Table 2 Comparison of Different Benchmarks Suites; Adopted from [70]

	Diverse characteristics	MineBench	GridMix	HiBench	WL suite	CloudSuite	CloudRank-D
Representative applications	Basic operations	No	Yes	Yes	Yes	No	Yes
	Classification	Yes	No	Yes	No	Yes	Yes
	Clustering	Yes	No	Yes	No	No	Yes
	Recommendation	No	No	No	No	No	Yes
	Sequence learning	Yes	No	No	No	No	Yes
	Association rule mining	Yes	No	No	No	No	Yes
	Data warehouse operations	No	No	No	No	No	Yes
Workloads description	Submission pattern	No	No	No	Yes	No	Yes
	Scheduling policies	No	No	No	No	No	Yes
	System configurations	No	No	No	No	No	Yes
	Data models	No	No	No	No	No	Yes
	Data semantics	No	No	No	No	No	Yes
	Multiple data inputs	Yes	Yes	No	Yes	No	Yes
	Category of data-centric computation	No	No	No	Yes	No	Yes

Table 3 Different targets and metrics among benchmark suites; adopted from [70]

	Targets	Metrics
MineBench	Data mining algorithm on single-node computers	No
GridMix	Hadoop framework	Number of jobs and running time
HiBench	Hadoop framework	Job running time, the number of tasks completed per minute
WL suite	Hadoop framework	No
CloudSuite	Architecture research	No
CloudRank-D	Evaluating cloud systems at the whole system level	Data processed per second and data processed per joule

the authors compare CloudRank-D with the listed benchmarks. This comparison is also not accurate, because the other benchmarks target only a specific technology such as Hadoop, or have no general metric as in the case of WL suites, MineBench and CloudSuite.

Chen et al. [71, 72] present their Statistical Workload Injector for MapReduce (SWIM), while investigating a number of important characteristics of MapReduce workloads, as part of which they compare the most popular Hadoop benchmarks. Based on this investigation, they identify two design goals used in SWIM:

(i) the workload synthesis and execution framework should be agnostic to hardware/software/configuration choices, cluster size, specific MapReduce implementation, and the underlying system; and
(ii) the framework should synthesize representative workloads with short duration of execution.

However, the proposed benchmark is specifically focused on analyzing the internal dynamics of pure MapReduce applications and how to generate representative synthetic data from real workloads. Apart from that, it does not perform other data and computational tasks typical for big data applications which benchmarks like HiBench and PigMix address. Also it does not report any metrics which can be used to compare the different systems under test.

A recent survey on benchmarks for big data [73] extensively reviews the current big data benchmarks and discusses some of the characteristics and challenges they should address. Table 4 summarizes this set of benchmarks. The benchmarks are compared only by reference to targeted platforms and main application characteristics. The list includes benchmarks such as TPC-C/H/W/DS and SSB, which target only a specific set of workload characteristics. The table looks more like a listing of benchmarks than a real comparison based on specific criteria. An important point that the authors make is the need of a more complete, end-to-end benchmarking suite, including both component-based and application-oriented benchmarks along with critical metrics such as energy consumption.

The paper presenting the BigDataBench suite by Wang et al. [74] also discusses extensively the challenges of developing a real big data benchmark and compares

Table 4 Comparison of existing works on big data benchmarks; Adopted from [73]

Work	Target	Characteristics	Comment
TPC-C	RDBMS	Transaction processing, simple query and update	OLTP
TPC-H	RDBMS, Hadoop Hive	Reporting, decision	OLAP
TPC-W	RDBMS, NoSQL	Web applications	Web OLTP
SSB	RDBMS, Hadoop Hive	Reporting, decision	OLAP
TPC-DS	RDBMS, Hadoop Hive	Reporting query, ad hoc query, iterative query, data mining query	OLAP
TeraSort	RDBMS, Hadoop	Data sorting	Sorting only
YCSB	NoSQL database	Cloud-based data serving	Web OLTP
REF 11	Unstructured data management system	Unstructured data only edge detection, proximity search, data scanning, data fusion	Not representative enough
GRAPH 500	Graph NoSQL database	Graph data processing only	Not representative enough
LinkBench	RDBMS, graph NoSQL database	Modeling Facebook real life application graph data processing only	Not representative enough
DFSIO	Hadoop	File system level benchmark	Not representative enough
Hive performance benchmark	Hadoop Hive	GREP, selection, aggregation, join and UDF aggregation only	Not representative enough
GridMix	Hadoop	Mix of Hadoop jobs	Not representative enough
PUMA	MapReduce	Term-vector, inverted-index, self-join, adjacency-list, k-means, classification, histogram-movies, histogram-ratings, sequence-count, ranked inverted index, Tera-sort, GREP, word-count	Comprehensive workload
MRBench	MapReduce	TPC-H queries	OLAP
HiBench	MapReduce	Micro-benchmarks (sort, word count and TeraSort); Web search (Nutch Indexing and page rank) machine learning (Bayesian classification and k-means clustering); HDFS benchmark (file system level benchmark)	Comprehensive workload
CloudRank-D	RDBMS, Hadoop	Basic operations for data analysis, classification, clustering, recommendation, sequence learning, association rule mining, and data warehouse queries	Comprehensive workload
BigBench	RDBMS, Hadoop	Covers data models of structured, semi-structured and unstructured data; addresses variety, velocity and volume aspects of big data systems	Comprehensive workload

Table 5 Comparison of big data benchmarking efforts; adopted from [74]

Benchmark Efforts	Real-world data sets (Data Set Number)	Data scalability (Volume, Veracity)	Workloads variety	Software stacks	Objects to test	Status
HiBench	Unstructured text data (1)	Partial	Offline Analytics; Real-time Analytics	Hadoop and Hive	Hadoop and Hive	Open source
BigBench	None	N/A	Offline Analytics	DBMS and Hadoop	DBMS and Hadoop	Proposal
AMP Benchmarks	None	N/A	Real-time Analytics	Real-time analytic systems	Real-time analytic systems	Open source
YCSB	None	N/A	Online Services	NoSQL systems	NoSQL systems	Open source
LinkBench	Unstructured graph data (1)	Partial	Online Services	Graph database	Graph database	Open source
CouldSuite	Unstructured text data (1)	Partial	Online Services; Offline Analytics	NoSQL systems, Hadoop, GraphLab	Architectures	Open source
BigDataBench	Unstructured text data (1) Semi-structured text data (1) Unstructured graph data (2) Structured table data (1) Semi-structured table data (1)	Total	Online Services; Offline Analytics; Real-time Analytics	NoSQL systems, DBMS, Real-time Analytics Offline Analytics systems	Systems and architecture; NoSQL systems; Different analytics systems	Open source

the BigDataBench with other existing suites. The resulting list, depicted in Table 5, compares them according to data sets included, data scalability, workload variety, software stack and status. Clearly, BigDataBench leads in all of the characteristics as its goal is to be an end-to-end benchmark, but the other benchmarks included, like AMP Benchmarks [75], YCSB [76] and LinkBench [77], have different functionalities and are platform specific.

In short, none of the above reviewed benchmark comparisons evaluates and categorizes the existing benchmarks suites in an objective way. In order for this to be done, an independent classification, targeting the entire spectrum of big data benchmark types and based on clearly set criteria, should be constructed.

4 Conclusions

In this chapter, we aimed to outline some of the main issues that are relevant when setting up and optimizing a big data project. We concentrated our attention first on the managerial task of setting up a big data project using insights from industry leaders, then we looked in some detail at the available data management and processing technologies for big data, and concluded by looking at the task of defining effective benchmarks for big data. Effective benchmarks for big data help the customers pick the optimal technology, help the vendors improve their products, and finally help researchers understand the differences of big data technologies on their path to optimize organizational and technical processes.

References

1. On Big Data Velocity. Interview with Scott Jarr, ODBMS Industry Watch, 28 Jan 2013. http://www.odbms.org/blog/2013/01/on-big-data-velocity-interview-with-scott-jarr/ (2015). Accessed 15 July 2015
2. How to run a Big Data project. Interview with James Kobielus. ODBMS Industry Watch, 15 May 2014. http://www.odbms.org/blog/2014/05/james-kobielus/ (2015). Accessed 15 July 2015
3. Laney, D.: 3D data management: controlling data volume, velocity and variety. Appl. Deliv. Strateg. File, **949** (2001)
4. Zikopoulos, P., Eaton, C.: Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data, 1st ed. McGraw-Hill Osborne Media (IBM) (2011)
5. Foster, I.: Big Process for Big Data, Presented at the HPC 2012 Conference. Cetraro, Italy (2012)
6. Gattiker, A., Gebara, F.H., Hofstee, H.P., Hayes, J.D., Hylick, A.: Big Data text-oriented benchmark creation for Hadoop. IBM J. Res. Dev., **57**(3/4), 10: 1–10: 6 (2013)
7. Zicari, R.: Big Data: Challenges and Opportunities. In: Akerkar, R. (ed.) Big Data Computing, p. 564. Chapman and Hall/CRC (2013)
8. On Big Data: Interview with Dr. Werner Vogels, CTO and VP of Amazon.com. ODBMS Industry Watch, 02 Nov 2011. http://www.odbms.org/blog/2011/11/on-big-data-interview-with-dr-werner-vogels-cto-and-vp-of-amazon-com/ (2015). Accessed 15 July 2015

9. Big Data Analytics at Thomson Reuters. Interview with Jochen L. Leidner. ODBMS Industry Watch, 15 Nov 2013. http://www.odbms.org/blog/2013/11/big-data-analytics-at-thomson-reuters-interview-with-jochen-l-leidner/ (2015). Accessed 15 July 2015
10. Setting up a Big Data project. Interview with Cynthia M. Saracco. ODBMS Industry Watch, 27 Jan 2014. http://www.odbms.org/blog/2014/01/setting-up-a-big-data-project-interview-with-cynthia-m-saracco/ (2015). Accessed 15 July 2015
11. Jacobs, A.: The pathologies of big data. Commun. ACM **52**(8), 36–44 (2009)
12. On Big Data and Hadoop. Interview with Paul C. Zikopoulos. ODBMS Industry Watch, 10 June 2013. http://www.odbms.org/blog/2013/06/on-big-data-and-hadoop-interview-with-paul-c-zikopoulos/ (2015). Accessed 15 July 2015
13. Next generation Hadoop. Interview with John Schroeder. ODBMS Industry Watch, 07 Sep 2012. http://www.odbms.org/blog/2012/09/next-generation-hadoop-interview-with-john-schroeder/ (2015). Accessed 15 July 2015
14. On Big Data, Analytics and Hadoop. Interview with Daniel Abadi. ODBMS Industry Watch, 05 Dec 2012. http://www.odbms.org/blog/2012/12/on-big-data-analytics-and-hadoop-interview-with-daniel-abadi/ (2015). Accessed 15 July 2015
15. Data Analytics at NBCUniversal. Interview with Matthew Eric Bassett. ODBMS Industry Watch, 23 Sep 2013. http://www.odbms.org/blog/2013/09/data-analytics-at-nbcuniversal-interview-with-matthew-eric-bassett/ (2015). Accessed 15 July 2015
16. Analytics: The real-world use of big data. How innovative enterprises extract value from uncertain data (IBM Institute for Business Value and Saïd Business School at the University of Oxford), Oct 2012
17. Hopkins, B.: The Patterns of Big Data. Forrester Research, 11 June 2013
18. Lim, H., Han, Y., Babu, S.: How to Fit when No One Size Fits. In: CIDR (2013)
19. Cattell, R.: Scalable SQL and NoSql Data Stores. SIGMOD Rec., **39**(4), 27 Dec 2010
20. Gilbert, S., Lynch, N.: Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News **33**(2), 51–59 (2002)
21. Haerder, T., Reuter, A.: Principles of transaction-oriented database recovery. ACM Comput. Surv. **15**(4), 287–317 (1983)
22. Bailis, P., Ghodsi, A.: Eventual Consistency Today: Limitations, Extensions, and Beyond. Queue **11**(3), pp. 20:20–20:32, Mar 2013
23. Pritchett, D.: BASE: an acid alternative. Queue **6**(3), 48–55 (2008)
24. Vogels, W.: Eventually consistent. Commun. ACM **52**(1), 40–44 (2009)
25. Moniruzzaman, A.B.M., Hossain, S.A.: NoSQL Database: New Era of Databases for Big data Analytics—Classification, Characteristics and Comparison. CoRR (2013). arXiv:1307.0191
26. Datastax, Datastax Apache Cassandra 2.0 Documentation. http://www.datastax.com/documentation/cassandra/2.0/index.html (2015). Accessed 15 Apr 2015
27. Apache Cassandra White Paper. http://www.datastax.com/wp-content/uploads/2011/02/DataStax-cBackgrounder.pdf
28. MongoDB Inc., MongoDB Documentation. http://docs.mongodb.org/manual/MongoDB-manual.pdf (2015). Accessed 15 Apr 2015
29. Chang, F., Dean, S., Ghemawat, W.C., Hsieh, D.A. Wallach, Burrows, M., Chandra, T., Fikes, A.,Gruber, R.E.:Bigtable: a distributed storage system for structured data. In: Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation, vol 7, pp. 15–15. Berkeley, CA, USA (2006)
30. George, L.: HBase: The Definitive Guide, 1st ed. O'Reilly Media (2011)
31. Apache Software Foundation, The Apache HBase Reference Guide. https://hbase.apache.org/book.html
32. Buerli, M.: The Current State of Graph Databases, Dec-2012, http://www.cs.utexas.edu/~cannata/dbms/Class%20Notes/08%20Graph_Databases_Survey.pdf (2015). Accessed 15 Apr 2015
33. Angles, R.: A comparison of current graph database models. In: ICDE Workshops, pp. 171–177 (2012)

34. McColl, R.C., Ediger, D., Poovey, J., Campbell, D., Bader, D.A.: A performance evaluation of open source graph databases. In: Proceedings of the First Workshop on Parallel Programming for Analytics Applications, pp. 11–18. New York, NY, USA (2014)
35. Harris, S., Seaborne, A.: SPARQL 1.1 Query Language. SPARQL 1.1 Query Language, 21-Mar-2013. http://www.w3.org/TR/sparql11-query/ (2013)
36. Holzschuher, F., Peinl, R.: Performance of graph query languages: comparison of cypher, gremlin and native access in Neo4 J. In: Proceedings of the Joint EDBT/ICDT 2013 Workshops, pp. 195–204. New York, NY, USA (2013)
37. VoltDB Inc., Using VoltDB. http://voltdb.com/download/documentation/
38. Pezzini, M., Edjlali, R.: Gartner top technology trends, 2013. In: Memory Computing Aims at Mainstream Adoption, 31 Jan 2013
39. Herschel, G., Linden, A., Kart, L.: Gartner Magic Quadrant for Advanced Analytics Platforms, 19 Feb 2014
40. Borthakur, D.: The hadoop distributed file system: Architecture and design. Hadoop Proj. Website **11**, 21 (2007)
41. Ghemawat, S., Gobioff, H., Leung, S.-T.: The google file system. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, pp. 29–43. New York, NY, USA (2003)
42. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. Commun. ACM **51**(1), 107–113 (2008)
43. Thusoo, A., Sarma, J.S., Jain, N., Shao, Z., Chakka, P., Anthony, S., Liu, H., Wyckoff, P., Murthy, R.: Hive: A Warehousing Solution over a Map-reduce Framework. Proc. VLDB Endow. **2**(2), 1626–1629 (2009)
44. Olston, C., Reed, B., Srivastava, U., Kumar, R., Tomkins, A.: Pig latin: a not-so-foreign language for data processing. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 1099–1110 (2008)
45. Shvachko, K., Kuang, H., Radia, S., Chansler, R.: The hadoop distributed file system. In: Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), pp. 1–10. Washington, DC, USA (2010)
46. White, T.: Hadoop: The Definitive Guide, 1st ed. O'Reilly Media, Inc., (2009)
47. Apache Spark Project. http://spark.apache.org/
48. Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., Stoica, I.: Spark: cluster computing with working sets. In: Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing, pp. 10–10. Berkeley, CA, USA (2010)
49. Cranor, C., Johnson, T., Spataschek, O., Shkapenyuk, V.: Gigascope: a stream database for network applications. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, pp. 647–651. New York, NY, USA (2003)
50. Arasu, A., Babu, S., Widom, J.: The CQL Continuous Query Language: Semantic Foundations and Query Execution. VLDB J. **15**(2), 121–142 (2006)
51. Chen, J., DeWitt, D.J., Tian, F., Wang, Y.: NiagaraCQ: a scalable continuous query system for internet databases. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 379–390. New York, NY, USA (2000)
52. Agrawal, J., Diao, Y., Gyllstrom, D, Immerman, N.: Efficient pattern matching over Event streams. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 147–160. New York, NY, USA (2008)
53. Jain, N., Mishra, S., Srinivasan, A., Gehrke, J., Widom, J., Balakrishnan, H., Çetintemel, U., Cherniack, M., Tibbetts, R., Zdonik, S.: Towards a Streaming SQL Standard. Proc VLDB Endow **1**(2), 1379–1390 (2008)
54. Balkesen, C., Tatbul, N.: Scalable data partitioning techniques for parallel sliding window processing over data streams. In: VLDB International Workshop on Data Management for Sensor Networks (DMSN'11). Seattle, WA, USA (2011)
55. Ahmad, Y., Berg, B., Çetintemel, U., Humphrey, M., Hwang, J.-H., Jhingran, A., Maskey, A., Papaemmanouil, O., Rasin, A., Tatbul, N., Xing, W., Xing, Y., Zdonik, S.B.: Distributed operation in the Borealis stream processing engine. In: SIGMOD Conference, pp. 882–884 (2006)

56. Apache Spark. http://spark.apache.org/
57. Apache Storm. http://storm.incubator.apache.org/
58. Amazon Kinesis. http://aws.amazon.com/kinesis/
59. Gualtieri, M., Curran, R.: The Forrester Wave: Big Data Streaming Analytics Platforms, Q3 2014, 17 July 2014
60. Tibco Streambase. http://www.streambase.com
61. Ivanov, T., Niemann, R., Izberovic, S., Rosselli, M., Tolle, K., Zicari, R.V.: Performance evaluationi of enterprise big data platforms with HiBench. presented at the In: 9th IEEE International Conference on Big Data Science and Engineering (IEEE BigDataSE 2015), Helsinki, Finland, 20–22 Aug 2015
62. Ivanov, T., Beer, M.: Performance evaluation of spark SQL using BigBench. Presented at the In: 6th Workshop on Big Data Benchmarking (6th WBDB). Canada, Toronto, 16–17 June 2015
63. Rosselli, M., Niemann, R., Ivanov, T., Tolle, K., Zicari, R.V.: "Benchmarking the Availability and Fault Tolerance of Cassandra", presented at the In 6th Workshop on Big Data Benchmarking (6th WBDB), June 16–17, 2015. Canada, Toronto (2015)
64. TPC, TPC-H - Homepage. http://www.tpc.org/tpch/ (2015). Accessed 15 July 2015
65. TPC Big Data Working Group, TPC-BD - Homepage TPC Big Data Working Group. http://www.tpc.org/tpcbd/default.asp (2015). Accessed 15 July 2015
66. BigData Top100, 2013. http://bigdatatop100.org/ (2015). Accessed 15 July 2015
67. Big Data Benchmarking Community, Big Data Benchmarking | Center for Large-scale Data Systems Research, Big Data Benchmarking Community. http://clds.ucsd.edu/bdbc/ (2015). Accessed 15 July 2015
68. Chen, Y.: We don't know enough to make a big data benchmark suite-an academia-industry view. Proc. WBDB (2012)
69. Xiong, W., Yu, Z., Bei, Z., Zhao, J., Zhang, F., Zou, Y., Bai, X., Li, Y., Xu, C.: A characterization of big data Benchmarks. In: Big Data. IEEE International Conference on **2013**, 118–125 (2013)
70. Luo, C., Zhan, J., Jia, Z., Wang, L., Lu, G., Zhang, L., Xu, C.-Z., Sun, N.: CloudRank-D: benchmarking and ranking cloud computing systems for data processing applications. Front. Comput. Sci. **6**(4), 347–362 (2012)
71. Chen, Y., Ganapathi, A., Griffith, R., Katz, R.: The case for evaluating MapReduce performance using workload suites. In: 2011 IEEE 19th International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 390–399 (2011)
72. Chen, Y., Alspaugh, S., Katz, R.: Interactive analytical processing in big data systems: a cross-industry study of MapReduce workloads. Proc. VLDB Endow. **5**(12), 1802–1813 (2012)
73. Qin, X., Zhou, X.: A survey on Benchmarks for big data and some more considerations. In: Intelligent Data Engineering and Automated Learning–IDEAL. Springer **2013**, 619–627 (2013)
74. Wang, L., Zhan, J., Luo, C., Zhu, Y, Yang, Q., He, Y., Gao, W., Jia, Z., Shi, Y., Zhang, S.: Bigdatabench: a big data benchmark suite from internet services. arXiv:14011406 (2014)
75. AMP Lab Big Data Benchmark. https://amplab.cs.berkeley.edu/benchmark/ (2015). Accessed 15 July 2015
76. Patil, S., Polte, M., Ren, K., Tantisiriroj, W., Xiao, L., López, J., Gibson, G., Fuchs, A., Rinaldi, B.: Ycsb ++: benchmarking and performance debugging advanced features in scalable table stores. In: Proceedings of the 2nd ACM Symposium on Cloud Computing, p. 9 (2011)
77. Armstrong, T.G., Ponnekanti, V., Borthakur, D., Callaghan, M.: Linkbench: a database benchmark based on the facebook social graph. In: Proceedings of the 2013 international conference on Management of data, pp. 1185–1196 (2013)

Author Biographies

Roberto V. Zicari is Full Professor of Database and Information Systems at Frankfurt University. He is an internationally recognized expert in the field of databases. His interests also expand to Innovation and Entrepreneurship. He is the founder of the Big Data Lab at the Goethe University Frankfurt, and the editor of ODBMS.org web portal and ODBMS Industry Watch Blog. He is also a visiting professor with the Center for Entrepreneurship and Technology within the Department of Industrial Engineering and Operations Research at UC Berkeley. Previously, Roberto served as associate professor at Politecnico di Milano, Italy; visiting scientist at IBM Almaden Research Center, USA, the University of California at Berkeley, USA; visiting professor at EPFL in Lausanne, Switzerland, the National University of Mexico City, Mexico and the Copenhagen Business School, Denmark.

Marten Rosselli is a Senior Consultant at Accenture and an expert for Big Data Management and Analytics. He has worked as a Technology Architect in many industrial projects in different industries such as Financial Services, Media and Telecommunication, Automotive, Public Services and Chemical. His expertise includes Project Management, Big Data Management and Analytics, Data Architectures, Software Engineering and Agile Software Development, Data Warehouse Technologies and Business Intelligence. He received his Master degree in Computer Science from the Goethe University Frankfurt am Main and he is a member and Ph.D. student at the Frankfurt Big Data Lab at the University of Frankfurt, Germany.

Todor Ivanov is a Ph.D. student at the Frankfurt Big Data Lab, Goethe University Frankfurt advised by Professor Roberto V. Zicari. He received his BSc. in Computational Engineering and MSc in Distributed Software Systems from Technical University of Darmstadt. His main interests are in the design and benchmarking of complex distributed software systems for big data and data-intensive applications.

Prior to that, he has worked as a senior software engineer developing Flight Information Display Systems (FIDS) for different international airports, research assistant in the FlashyDB project at the department of Databases and Distributed Systems, TU Darmstadt and IT consultant in business intelligence and database related projects in the bank sector.

Nikolaos Korfiatis is Assistant Professor of Business Analytics and Regulation at Norwich Business School, University of East Anglia (UEA) and faculty member at the Centre for Competition Policy (CCP).He is a an affiliate faculty member at the Big Data Laboratory Frankfurt where he previously acted as co-director responsible for Data Science and Analytics. He is active in the industry as a Senior Data Scientist in Residence for Adastra Germany, developing practical use cases for analytics and big data for the automotive and banking sector. He received his Ph.D. in Information Management at Copenhagen Business School (CBS) Denmark and his Diploma and MSc in Engineering (Specialization Information Retrieval) from Royal Institute of Technology (KTH), Stockholm and Uppsala University.

Karsten Tolle currently holds the position of an "Akademischer Rat" (Assistant Professor) at the Database and Information Systems (DBIS) group of the Goethe University Frankfurt. His current roles and activities include: Managing director of the Frankfurt Big Data Lab; Member of the Nomisma.org steering committee; Chair of the OASIS UBL German Localization Subcommittee. His main interests are in Linked Open Data (LOD) and Graph Databases. He studied mathematics with computer science as a special subject at the University of Hannover. He wrote his Master's Thesis during a SOCRATES-ERASMUS exchange program on Crete (Greece) at the ICS-Foundation of Research and Technology - Hellas (FORTH) and received his Ph.D. from the University of Frankfurt (Germany) in the area of Semantic Web.

Raik Niemann studied computer science at the University of Rostock and at the University of Applied Science Stralsund, Germany. He received a german Dipl.-Ing. (FH) and Master of Science degree. Currently Mr. Niemann is a Ph.D. student at the DBIS (Goethe University Frankfurt/Main) and focuses on the energy efficiency of data management systems with very large datasets. This includes all kinds data organization schemes (relational, object-relational, NoSQL and NewSQL), their performance in terms of response time as well as technical architectures and settings.

Christoph Reichenbach is junior professor for Software Engineering and Programming Languages at Goethe University Frankfurt. His principal expertise is in domain-specific languages, software tools, language run-time systems, and compilers. He received his Ph.D. from the University of Colorado at Boulder, and has previously held the positions of software engineer for search quality at Google and postdoctoral fellow at the University of Massachusetts, Amherst.

http://www.springer.com/978-3-319-30263-8

Big Data Optimization: Recent Developments and Challenges

Emrouznejad, A. (Ed.)
2016, XV, 487 p. 182 illus., 160 illus. in color.,
Hardcover
ISBN: 978-3-319-30263-8