Massively Parallel Databases and MapReduce Systems

Shivnath Babu
Duke University
shivnath@cs.duke.edu

Herodotos Herodotou
Microsoft Research
herohero@microsoft.com

the essence of knowledge
Boston — Delft

Foundations and Trends® in Databases

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

S. Babu and H. Herodotou. *Massively Parallel Databases and MapReduce Systems.* Foundations and Trends® in Databases, vol. 5, no. 1, pp. 1–104, 2012.

This Foundations and Trends® issue was typeset in LaTeX using a class file designed by Neal Parikh. Printed on acid-free paper.

ISBN: 978-1-60198-758-1

Foundations and Trends® in Databases
Volume 5, Issue 1, 2012

Editorial Scope

Topics

Foundations and Trends® in Databases covers a breadth of topics relating to the management of large volumes of data. The journal targets the full scope of issues in data management, from theoretical foundations, to languages and modeling, to algorithms, system architecture, and applications. The list of topics below illustrates some of the intended coverage, though it is by no means exhaustive:

- Data models and query languages
- Query processing and optimization
- Storage, access methods, and indexing
- Transaction management, concurrency control, and recovery
- Deductive databases
- Parallel and distributed database systems
- Database design and tuning
- Metadata management
- Object management
- Trigger processing and active databases
- Data mining and OLAP
- Approximate and interactive query processing
- Data warehousing
- Adaptive query processing
- Data stream management
- Search and query integration
- XML and semi-structured data
- Web services and middleware
- Data integration and exchange
- Private and secure data management
- Peer-to-peer, sensornet, and mobile data management
- Scientific and spatial data management
- Data brokering and publish/subscribe
- Data cleaning and information extraction
- Probabilistic data management

Information for Librarians

Foundations and Trends® in Databases, 2012, Volume 5, 4 issues. ISSN paper version 1931-7883. ISSN online version 1931-7891. Also available as a combined paper and online subscription.

Foundations and Trends® in Databases
Vol. 5, No. 1 (2012) 1–104

DOI: 10.1561/1900000036

the essence of knowledge

Massively Parallel Databases and MapReduce Systems

Shivnath Babu
Duke University
shivnath@cs.duke.edu

Herodotos Herodotou
Microsoft Research
herohero@microsoft.com

Contents

Abstract

Timely and cost-effective analytics over "big data" has emerged as a key ingredient for success in many businesses, scientific and engineering disciplines, and government endeavors. Web clicks, social media, scientific experiments, and datacenter monitoring are among data sources that generate vast amounts of raw data every day. The need to convert this raw data into useful information has spawned considerable innovation in systems for large-scale data analytics, especially over the last decade. This monograph covers the design principles and core features of systems for analyzing very large datasets using massively-parallel computation and storage techniques on large clusters of nodes. We first discuss how the requirements of data analytics have evolved since the early work on parallel database systems. We then describe some of the major technological innovations that have each spawned a distinct category of systems for data analytics. Each unique system category is described along a number of dimensions including data model and query interface, storage layer, execution engine, query optimization, scheduling, resource management, and fault tolerance. We conclude with a summary of present trends in large-scale data analytics.

S. Babu and H. Herodotou. *Massively Parallel Databases and MapReduce Systems*. Foundations and Trends® in Databases, vol. 5, no. 1, pp. 1–104, 2012.

DOI: 10.1561/1900000036.

1

Introduction

Organizations have always experienced the need to run data analytics tasks that convert large amounts of raw data into the information required for timely decision making. Parallel databases like Gamma [75] and Teradata [188] were some of the early systems to address this need. Over the last decade, more and more sources of large datasets have sprung up, giving rise to what is popularly called *big data*. Web clicks, social media, scientific experiments, and datacenter monitoring are among such sources that generate vast amounts of data every day.

Rapid innovation and improvements in productivity necessitate timely and cost-effective analysis of big data. This need has led to considerable innovation in systems for large-scale data analytics over the last decade. Parallel databases have added techniques like columnar data storage and processing [39, 133]. Simultaneously, new distributed compute and storage systems like MapReduce [73] and Bigtable [58] have been developed. This monograph is an attempt to cover the design principles and core features of systems for analyzing very large datasets. We focus on systems for large-scale data analytics, namely, the field that is called Online Analytical Processing (OLAP) as opposed to Online Transaction Processing (OLTP).

We begin in this chapter with an overview of how we have organized the overall content. The overview first discusses how the requirements of data analytics have evolved since the early work on parallel database systems. We then describe some of the major technological innovations that have each spawned a distinct category of systems for data analytics. The last part of the overview describes a number of dimensions along which we will describe and compare each of the categories of systems for large-scale data analytics.

The overview is followed by four chapters that each discusses one unique category of systems in depth. The content in the following chapters is organized based on the dimensions that will be identified in this chapter. We then conclude with a summary of present trends in large-scale data analytics.

1.1 Requirements of Large-scale Data Analytics

The Classic Systems Category: Parallel databases—which constitute the *classic* system category that we discuss—were the first systems to make parallel data processing available to a wide class of users through an intuitive high-level programming model. Parallel databases were based predominantly on the relational data model. The declarative SQL was used as the query language for expressing data processing tasks over data stored as tables of records.

Parallel databases achieved high performance and scalability by partitioning tables across the nodes in a shared-nothing cluster. Such a horizontal partitioning scheme enabled relational operations like filters, joins, and aggregations to be run in parallel over different partitions of each table stored on different nodes.

Three trends started becoming prominent in the early 2000s that raised questions about the superiority of classic parallel databases:

- More and more companies started to store as much data as they could collect. The classic parallel databases of the day posed major hurdles in terms of scalability and total cost of ownership as the need to process these ever-increasing data volumes arose.
- The data being collected and stored by companies was diverse in

structure. For example, it became a common practice to collect highly structured data such as sales data and user demographics along with less structured data such as search query logs and web page content. It was hard to fit such diverse data into the rigid data models supported by classic parallel databases.

- Business needs started to demand shorter and shorter intervals between the time when data was collected (typically in an OLTP system) and the time when the results of analyzing the data were available for manual or algorithmic decision making.

These trends spurred two types of innovations: (a) innovations aimed at addressing the deficiencies of classic parallel databases while preserving their strengths such as high performance and declarative query languages, and (b) innovations aimed at creating alternate system architectures that can support the above trends in a cost-effective manner. These innovations, together with the category of classic parallel database systems, give the four unique system categories for large-scale data analytics that we will cover. Table 1.1 lists the system categories and some of the systems that fall under each category.

1.2 Categorization of Systems

The Columnar Systems Category: Columnar systems pioneered the concept of storing tables by collocating entire columns together instead of collocating rows as done in classic parallel databases. Systems with columnar storage and processing, such as Vertica [133], have been shown to use CPU, memory, and I/O resources more efficiently in large-scale data analytics compared to row-oriented systems. Some of the main benefits come from reduced I/O in columnar systems by reading only the needed columns during query processing. Columnar systems are covered in Chapter 3.

The MapReduce Systems Category: MapReduce is a programming model and an associated implementation of a run-time system that was developed by Google to process massive datasets by harnessing a very large cluster of commodity nodes [73]. Systems in the classic

Category	Example Systems in this Category
Classic	Aster nCluster [25, 92], DB2 Parallel Edition [33], Gamma [75], Greenplum [99], Netezza [116], SQL Server Parallel Data Warehouse [177], Teradata [188]
Columnar	Amazon RedShift [12], C-Store [181], Infobright [118], MonetDB [39], ParAccel [164], Sybase IQ [147], VectorWise [206], Vertica [133]
MapReduce	Cascading [52], Clydesdale [123], Google MapReduce [73], Hadoop [192, 14], HadoopDB [5], Hadoop++ [80], Hive [189], JAQL [37], Pig [94]
Dataflow	Dremel [153], Dryad [197], Hyracks [42], Nephele [34], Pregel [148], SCOPE [204], Shark [195], Spark [199]

Table 1.1: The system categories that we consider, and some of the systems that fall under each category.

category have traditionally struggled to scale to such levels. MapReduce systems pioneered the concept of building multiple standalone scalable distributed systems, and then composing two or more of these systems together in order to run analytic tasks on large datasets. Popular systems in this category, such as Hadoop [14], store data in a standalone block-oriented distributed file-system, and run computational tasks in another distributed system that supports the MapReduce programming model. MapReduce systems are covered in Chapter 4.

The Dataflow Systems Category: Some deficiencies in MapReduce systems were identified as these systems were used for a large number of data analysis tasks. The MapReduce programming model is too restrictive to express certain data analysis tasks easily, e.g., joining two datasets together. More importantly, the execution techniques used by MapReduce systems are suboptimal for many common types of data analysis tasks such as relational operations, iterative machine learning, and graph processing. Most of these problems can be addressed by replacing MapReduce with a more flexible dataflow-based execution model that can express a wide range of data access and communication patterns. Various dataflow-based execution models have been used by the systems in this category, including directed acyclic graphs in Dryad [197], serving trees in Dremel [153], and bulk synchronous parallel processing in Pregel [148]. Dataflow systems are covered in Chapter 5.

Other System Categories: It became clear over time that new systems can be built by combining design principles from different system categories. For example, techniques used for high-performance processing in classic parallel databases can be used together with techniques used for fine-grained fault tolerance in MapReduce systems [5]. Each system in this *coalesced* category exposes a unified system interface that provides a combined set of features that are traditionally associated with different system categories. We will discuss coalesced systems along with the other system categories in the respective chapters.

The need to reduce the gap between the generation of data and the generation of analytics results over this data has required system developers to constantly raise the bar in large-scale data analytics. On one hand, this need saw the emergence of scalable distributed storage systems that provide various degrees of transactional capabilities. Support for transactions enables these systems to serve as the data store for online services while making the data available concurrently in the same system for analytics. The same need has led to the emergence of parallel database systems that support both OLTP and OLAP in a single system. We put both types of systems into the category called *mixed* systems because of their ability to run mixed workloads—workloads that contain transactional as well as analytics tasks—efficiently. We will discuss mixed systems in Chapter 6 as part of recent trends in massively parallel data analytics.

1.3 Categorization of System Features

We have selected eight key system features along which we will describe and compare each of the four categories of systems for large-scale data analytics.

Data Model and Interfaces: A *data model* provides the definition and logical structure of the data, and determines in which manner data can be stored, organized, and manipulated by the system. The most popular example of a data model is the relational model (which uses a table-based format), whereas most systems in the MapReduce and Dataflow categories permit data to be in any arbitrary format stored in flat files. The data model used by each system is closely related to the *query interface* exposed by the system, which allows users to manage and manipulate the stored data.

Storage Layer: At a high level, a *storage layer* is simply responsible for persisting the data as well as providing methods for accessing and modifying the data. However, the design, implementation and features provided by the storage layer used by each of the different system categories vary greatly, especially as we start comparing systems across the different categories. For example, classic parallel databases use integrated and specialized data stores that are tightly coupled with their execution engines, whereas MapReduce systems typically use an independent distributed file-system for accessing data.

Execution Engine: When a system receives a query for execution, it will typically convert it into an *execution plan* for accessing and processing the query's input data. The *execution engine* is the entity responsible for actually running a given execution plan in the system and generating the query result. In the systems that we consider, the execution engine is also responsible for parallelizing the computation across large-scale clusters of machines, handling machine failures, and setting up inter-machine communication to make efficient use of the network and disk bandwidth.

Query Optimization: In general, *query optimization* is the process a system uses to determine the most efficient way to execute a given query by considering several alternative, yet equivalent, execution plans. The techniques used for query optimization in the systems we consider are very different in terms of: (i) the space of possible execution plans (e.g., relational operators in databases versus configuration parameter settings in MapReduce systems), (ii) the type of query optimization (e.g., cost-based versus rule-based), (iii) the type of cost modeling technique (e.g., analytical models versus models learned using machine-learning techniques), and (iv) the maturity of the optimization techniques (e.g., fully automated versus manual tuning).

Scheduling: Given the distributed nature of most data analytics systems, *scheduling* the query execution plan is a crucial part of the system. Systems must now make several scheduling decisions, including scheduling where to run each computation, scheduling inter-node data transfers, as well as scheduling rolling updates and maintenance tasks.

Resource Management: *Resource management* primarily refers to the efficient and effective use of a cluster's resources based on the resource requirements of the queries or applications running in the system. In addition, many systems today offer elastic properties that allow users to dynamically add or remove resources as needed according to workload requirements.

Fault Tolerance: Machine failures are relatively common in large clusters. Hence, most systems have built-in *fault tolerance* functionalities that would allow them to continue providing services, possibly with graceful degradation, in the face of undesired events like hardware failures, software bugs, and data corruption. Examples of typical fault tolerance features include restarting failed tasks either due to application or hardware failures, recovering data due to machine failure or corruption, and using speculative execution to avoid stragglers.

System Administration: *System administration* refers to the activities where additional human effort may be needed to keep the system running smoothly while the system serves the needs of multiple users and applications. Common activities under system administration include performance monitoring and tuning, diagnosing the cause of poor performance or failures, capacity planning, and system recovery from permanent failures (e.g., failed disks) or disasters.

1.4 Related Work

This monograph is related to a few surveys done in the past. Lee and others have done a recent survey that focuses on parallel data processing with MapReduce [136]. In contrast, we provide a more comprehensive and in-depth coverage of systems for large-scale data analytics, and also define a categorization of these systems. Empirical comparisons have been done in the literature among different systems that we consider. For example, Pavlo and others have compared the performance of both classic parallel databases and columnar databases with the performance of MapReduce systems [166].

Tutorials and surveys have appeared in the past on specific dimensions along which we describe and compare each of the four categories of systems for large-scale data analytics. Recent tutorials include one on data layouts and storage in MapReduce systems [79] and one on programming techniques for MapReduce systems [174]. Kossmann's survey on distributed query processing [128] and Lu's survey on query processing in classic parallel databases [142] are also related.

References

[1] Daniel J. Abadi, Yanif Ahmad, Magdalena Balazinska, Ugur Çetintemel, Mitch Cherniack, Jeong-Hyon Hwang, Wolfgang Lindner, Anurag Maskey, Alex Rasin, Esther Ryvkina, Nesime Tatbul, Ying Xing, and Stanley B. Zdonik. The Design of the Borealis Stream Processing Engine. In *Proc. of the 3rd Biennial Conf. on Innovative Data Systems Research*, 2005.

[2] Daniel J. Abadi, Peter A. Boncz, and Stavros Harizopoulos. Column Oriented Database Systems. *Proc. of the VLDB Endowment*, 2(2):1664–1665, 2009.

[3] Daniel J Abadi, Don Carney, Ugur Çetintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Michael Stonebraker, Nesime Tatbul, and Stan Zdonik. Aurora: A New Model and Architecture for Data Stream Management. *The VLDB Journal*, 12(2):120–139, 2003.

[4] Daniel J. Abadi, Daniel S. Myers, David J. DeWitt, and Samuel Madden. Materialization Strategies in a Column-Oriented DBMS. In *Proc. of the 23rd IEEE Intl. Conf. on Data Engineering*, pages 466–475. IEEE, 2007.

[5] Azza Abouzeid, Kamil Bajda-Pawlikowski, Daniel Abadi, Alexander Rasin, and Avi Silberschatz. HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads. *Proc. of the VLDB Endowment*, 2(1):922–933, 2009.

[6] Aditya Agarwal, Mark Slee, and Marc Kwiatkowski. *Thrift: Scalable Cross-Language Services Implementation*, 2007. `http://thrift.apache.org/static/files/thrift-20070401.pdf`.

[7] Sameer Agarwal, Barzan Mozafari, Aurojit Panda, Henry Milner, Samuel Madden, and Ion Stoica. BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data. In *Proc. of the 8th European Conf. on Computer Systems*, pages 29–42. ACM, 2013.

[8] Sanjay Agrawal, Eric Chu, and Vivek Narasayya. Automatic Physical Design Tuning: Workload as a Sequence. In *Proc. of the 2006 ACM SIGMOD Intl. Conf. on Management of Data*, pages 683–694. ACM, 2006.

[9] Sanjay Agrawal, Vivek Narasayya, and Beverly Yang. Integrating Vertical and Horizontal Partitioning into Automated Physical Database Design. In *Proc. of the 2004 ACM SIGMOD Intl. Conf. on Management of Data*, pages 359–370. ACM, 2004.

[10] Anastassia Ailamaki, David J DeWitt, Mark D Hill, and Marios Skounakis. Weaving Relations for Cache Performance. *The VLDB Journal*, 1:169–180, 2001.

[11] Alexander Alexandrov, Max Heimel, Volker Markl, Dominic Battré, Fabian Hueske, Erik Nijkamp, Stephan Ewen, Odej Kao, and Daniel Warneke. Massively Parallel Data Analysis with PACTs on Nephele. *Proc. of the VLDB Endowment*, 3(1-2):1625–1628, 2010.

[12] *Amazon RedShift*, 2013. `http://aws.amazon.com/redshift/`.

[13] *Amazon Simple Storage Service (S3)*, 2013. `http://aws.amazon.com/s3/`.

[14] *Apache Hadoop*, 2012. `http://hadoop.apache.org/`.

[15] *Apache Hadoop Capacity Scheduler*, 2013. `http://hadoop.apache.org/docs/r1.1.2/capacity_scheduler.html`.

[16] *Apache Hadoop Fair Scheduler*, 2013. `http://hadoop.apache.org/docs/r1.1.2/fair_scheduler.html`.

[17] *Apache Hadoop on Demand*, 2013. `http://hadoop.apache.org/docs/stable/hod_scheduler.html`.

[18] *Apache Accumulo*, 2013. `http://accumulo.apache.org/`.

[19] *Apache Avro*, 2013. `http://avro.apache.org/`.

[20] *Apache Cassandra*, 2013. `http://cassandra.apache.org/`.

[21] *Apache Drill*, 2013. `http://incubator.apache.org/drill/`.

[22] *Apache Hadoop NextGen MapReduce (YARN)*, 2013. `http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html`.

[23] *Apache HBase*, 2013. `http://hbase.apache.org/`.

[24] *Apache HCatalog*, 2013. `http://incubator.apache.org/hcatalog/`.

[25] *Aster Data nCluster*, 2012. `http://www.asterdata.com/product/ncluster_cloud.php`.

[26] M. M. Astrahan, M. W. Blasgen, D. D. Chamberlin, K. P. Eswaran, J. N. Gray, P. P. Griffiths, W. F. King, R. A. Lorie, P. R. McJones, J. W. Mehl, G. R. Putzolu, I. L. Traiger, B. W. Wade, and V. Watson. System R: Relational Approach to Database Management. *ACM Trans. on Database Systems (TODS)*, 1(2):97–137, 1976.

[27] Shivnath Babu. Towards Automatic Optimization of MapReduce Programs. In *Proc. of the 1st Symp. on Cloud Computing*, pages 137–142. ACM, 2010.

[28] Shivnath Babu and Jennifer Widom. Continuous Queries over Data Streams. *ACM SIGMOD Record*, 30(3):109–120, 2001.

[29] Lakshmi N. Bairavasundaram, Garth R. Goodson, Bianca Schroeder, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. An Analysis of Data Corruption in the Storage Stack. *ACM Trans. on Storage (TOS)*, 4(3):8, 2008.

[30] Kamil Bajda-Pawlikowski, Daniel J Abadi, Avi Silberschatz, and Erik Paulson. Efficient Processing of Data Warehousing Queries in a Split Execution Environment. In *Proc. of the 2011 ACM SIGMOD Intl. Conf. on Management of Data*, pages 1165–1176. ACM, 2011.

[31] Jason Baker, Chris Bond, James Corbett, J. J. Furman, Andrey Khorlin, James Larson, Jean-Michel Leon, Yawei Li, Alexander Lloyd, and Vadim Yushprakh. Megastore: Providing Scalable, Highly Available Storage for Interactive Services. In *Proc. of the 5th Biennial Conf. on Innovative Data Systems Research*, pages 223–234, 2011.

[32] Ronald Barber, Peter Bendel, Marco Czech, Oliver Draese, Frederick Ho, Namik Hrle, Stratos Idreos, Min-Soo Kim, Oliver Koeth, Jae-Gil Lee, Tianchao Tim Li, Guy M. Lohman, Konstantinos Morfonios, René Müller, Keshava Murthy, Ippokratis Pandis, Lin Qiao, Vijayshankar Raman, Richard Sidle, Knut Stolze, and Sandor Szabo. Business Analytics in (a) Blink. *IEEE Data Engineering Bulletin*, 35(1):9–14, 2012.

[33] C. K. Baru, G. Fecteau, A. Goyal, H. Hsiao, A. Jhingran, S. Padmanabhan, G. P. Copeland, and W. G. Wilson. DB2 Parallel Edition. *IBM Systems Journal*, 34(2):292–322, 1995.

[34] Dominic Battré, Stephan Ewen, Fabian Hueske, Odej Kao, Volker Markl, and Daniel Warneke. Nephele/PACTs: A Programming Model and Execution Framework for Web-scale Analytical Processing. In *Proc. of the 1st Symp. on Cloud Computing*, pages 119–130. ACM, 2010.

[35] Alexander Behm, Vinayak R Borkar, Michael J Carey, Raman Grover, Chen Li, Nicola Onose, Rares Vernica, Alin Deutsch, Yannis Papakonstantinou, and Vassilis J Tsotras. ASTERIX: Towards a Scalable, Semistructured Data Platform for Evolving-world Models. *Distributed and Parallel Databases*, 29(3):185–216, 2011.

[36] K. Beyer, V. Ercegovac, and E. Shekita. *Jaql: A JSON query language*. `http://www.jaql.org`.

[37] Kevin S Beyer, Vuk Ercegovac, Rainer Gemulla, Andrey Balmin, Mohamed Eltabakh, Carl-Christian Kanne, Fatma Ozcan, and Eugene J Shekita. Jaql: A Scripting Language for Large Scale Semistructured Data Analysis. *Proc. of the VLDB Endowment*, 4(12):1272–1283, 2011.

[38] Alain Biem, Eric Bouillet, Hanhua Feng, Anand Ranganathan, Anton Riabov, Olivier Verscheure, Haris Koutsopoulos, and Carlos Moran. IBM Infosphere Streams for Scalable, Real-time, Intelligent Transportation Services. In *Proc. of the 2010 ACM SIGMOD Intl. Conf. on Management of Data*, pages 1093–1104. ACM, 2010.

[39] Peter Boncz, Torsten Grust, Maurice van Keulen, Stefan Manegold, Jan Rittinger, and Jens Teubner. MonetDB/XQuery: A Fast XQuery Processor Powered by a Relational Engine. In *Proc. of the 2006 ACM SIGMOD Intl. Conf. on Management of Data*, pages 479–490. ACM, 2006.

[40] Peter A. Boncz, Marcin Zukowski, and Niels Nes. MonetDB/X100: Hyper-Pipelining Query Execution. In *Proc. of the 2nd Biennial Conf. on Innovative Data Systems Research*, pages 225–237, 2005.

[41] H. Boral, W. Alexander, L. Clay, G. Copeland, S. Danforth, M. Franklin, B. Hart, M. Smith, and P. Valduriez. Prototyping Bubba, a Highly Parallel Database System. *IEEE Trans. on Knowledge and Data Engineering*, 2(1):4–24, 2002.

[42] Vinayak Borkar, Michael Carey, Raman Grover, Nicola Onose, and Rares Vernica. Hyracks: A Flexible and Extensible Foundation for Data-intensive Computing. In *Proc. of the 27th IEEE Intl. Conf. on Data Engineering*, pages 1151–1162. IEEE, 2011.

[43] Andrea J. Borr. Transaction Monitoring in ENCOMPASS: Reliable Distributed Transaction Processing. In *Proc. of the 7th Intl. Conf. on Very Large Data Bases*, pages 155–165, 1981.

[44] Dhruba Borthakur, Jonathan Gray, Joydeep Sen Sarma, Kannan Muthukkaruppan, Nicolas Spiegelberg, Hairong Kuang, Karthik Ranganathan, Dmytro Molkov, Aravind Menon, Samuel Rash, Rodrigo Schmidt, and Amitanand S. Aiyer. Apache Hadoop Goes Realtime at Facebook. In *Proc. of the 2011 ACM SIGMOD Intl. Conf. on Management of Data*, pages 1071–1080. ACM, 2011.

[45] Bobby-Joe Breitkreutz, Chris Stark, Mike Tyers, et al. Osprey: A Network Visualization System. *Genome Biol*, 4(3):R22, 2003.

[46] Kurt P. Brown, Manish Mehta, Michael J. Carey, and Miron Livny. Towards Automated Performance Tuning for Complex Workloads. In *Proc. of the 20th Intl. Conf. on Very Large Data Bases*, pages 72–84. VLDB Endowment, 1994.

[47] Yingyi Bu, Bill Howe, Magdalena Balazinska, and Michael Ernst. HaLoop: Efficient Iterative Data Processing on Large Clusters. *Proc. of the VLDB Endowment*, 3(1-2):285–296, 2010.

[48] Ron Buck. The Oracle Media Server for nCUBE Massively Parallel Systems. In *Proc. of the 8th Intl. Parallel Processing Symposium*, pages 670–673. IEEE, 1994.

[49] Michael J. Cafarella and Christopher Ré. Manimal: Relational Optimization for Data-Intensive Programs. In *Proc. of the 13th Intl. Workshop on the Web and Databases*, pages 10:1–10:6. ACM, 2010.

[50] Brad Calder, Ju Wang, Aaron Ogus, Niranjan Nilakantan, Arild Skjolsvold, Sam McKelvie, Yikang Xu, Shashwat Srivastav, Jiesheng Wu, Huseyin Simitci, et al. Windows Azure Storage: A Highly Available Cloud Storage Service with Strong Consistency. In *Proc. of the 23rd ACM Symp. on Operating Systems Principles*, pages 143–157. ACM, 2011.

[51] Michael J. Carey, Miron Livny, and Hongjun Lu. Dynamic Task Allocation in a Distributed Database System. In *Proc. of the 5th Intl. Conf. on Distributed Computing Systems*, pages 282–291. IEEE, 1985.

[52] *Cascading*, 2011. `http://www.cascading.org/`.

[53] Stefano Ceri, Mauro Negri, and Giuseppe Pelagatti. Horizontal Data Partitioning in Database Design. In *Proc. of the 1982 ACM SIGMOD Intl. Conf. on Management of Data*, pages 128–136, 1982.

[54] Ronnie Chaiken, Bob Jenkins, Per-Åke Larson, Bill Ramsey, Darren Shakib, Simon Weaver, and Jingren Zhou. SCOPE: Easy and Efficient Parallel Processing of Massive Data Sets. *Proc. of the VLDB Endowment*, 1(2):1265–1276, 2008.

[55] Craig Chambers, Ashish Raniwala, Frances Perry, Stephen Adams, Robert R. Henry, Robert Bradshaw, and Nathan Weizenbaum. FlumeJava: Easy, Efficient Data-parallel Pipelines. In *Proc. of the 2010 ACM SIGPLAN Conf. on Programming Language Design and Implementation*, pages 363–375, 2010.

[56] Badrish Chandramouli, Jonathan Goldstein, and Songyun Duan. Temporal Analytics on Big Data for Web Advertising. In *Proc. of the 28th IEEE Intl. Conf. on Data Engineering*, pages 90–101. IEEE, 2012.

[57] Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J Franklin, Joseph M Hellerstein, Wei Hong, Sailesh Krishnamurthy, Samuel R Madden, Fred Reiss, and Mehul A Shah. TelegraphCQ: Continuous Dataflow Processing. In *Proc. of the 2003 ACM SIGMOD Intl. Conf. on Management of Data*, pages 668–668. ACM, 2003.

[58] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. Bigtable: A Distributed Storage System for Structured Data. *ACM Trans. on Computer Systems*, 26(2):4, 2008.

[59] Biswapesh Chattopadhyay, Liang Lin, Weiran Liu, Sagar Mittal, Prathyusha Aragonda, Vera Lychagina, Younghee Kwon, and Michael Wong. Tenzing: A SQL Implementation on the MapReduce Framework. *Proc. of the VLDB Endowment*, 4(12):1318–1327, 2011.

[60] Surajit Chaudhuri, Arnd Christian König, and Vivek R. Narasayya. SQLCM: A Continuous Monitoring Framework for Relational Database Engines. In *Proc. of the 20th IEEE Intl. Conf. on Data Engineering*, pages 473–484, 2004.

[61] Chandra Chekuri, Waqar Hasan, and Rajeev Motwani. Scheduling Problems in Parallel Query Optimization. In *Proc. of the 14th ACM Symp. on Principles of Database Systems*, pages 255–265. ACM, 1995.

[62] Jianjun Chen, David J. DeWitt, Feng Tian, and Yuan Wang. NiagaraCQ: A Scalable Continuous Query System for Internet Databases. In *Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data*, pages 379–390. ACM, 2000.

[63] Ming-Syan Chen and Philip S. Yu. Interleaving a Join Sequence with Semijoins in Distributed Query Processing. *IEEE Trans. on Parallel Distributed Systems*, 3(5):611–621, 1992.

[64] Songting Chen. Cheetah: A High Performance, Custom Data Warehouse on Top of MapReduce. *Proc. of the VLDB Endowment*, 3(2):1459–1468, 2010.

[65] *Cloudera Impala*, 2013. `http://www.cloudera.com/content/cloudera/en/products/cdh/impala.html`.

[66] Richard Cole, Florian Funke, Leo Giakoumakis, Wey Guy, Alfons Kemper, Stefan Krompass, Harumi A. Kuno, Raghunath Othayoth Nambiar, Thomas Neumann, Meikel Poess, Kai-Uwe Sattler, Michael Seibold, Eric Simon, and Florian Waas. The Mixed Workload CH-benCHmark. In *Proc. of the 4th Intl. Workshop on Testing Database Systems*. ACM, 2011.

[67] Tyson Condie, Neil Conway, Peter Alvaro, Joseph M. Hellerstein, Khaled Elmeleegy, and Russell Sears. MapReduce Online. In *Proc. of the 7th USENIX Symp. on Networked Systems Design and Implementation*, volume 10. USENIX Association, 2010.

[68] George P. Copeland and Setrag Khoshafian. A Decomposition Storage Model. In *Proc. of the 1985 ACM SIGMOD Intl. Conf. on Management of Data*, pages 268–279, 1985.

[69] James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, JJ Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, Wilson Hsieh, Sebastian Kanthak, Eugene Kogan, Hongyi Li, Alexander Lloyd, Sergey Melnik, David Mwaura, David Nagle, Sean Quinlan, Rajesh Rao, Lindsay Rolig, Yasushi Saito, Michal Szymaniak, Christopher Taylor, Ruth Wang, and Dale Woodford. Spanner: Google's Globally-distributed Database. In *Proc. of the 10th USENIX Symp. on Operating Systems Design and Implementation*, page 1. USENIX Association, 2012.

[70] W Bruce Croft, Donald Metzler, and Trevor Strohman. *Search Engines: Information Retrieval in Practice*. Addison-Wesley Reading, 2010.

[71] Carlo Curino, Hyun Jin Moon, Alin Deutsch, and Carlo Zaniolo. Automating the Database Schema Evolution Process. *The VLDB Journal*, 22(1):73–98, 2013.

[72] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In *Proc. of the 6th USENIX Symp. on Operating Systems Design and Implementation*, pages 137–149. USENIX Association, 2004.

[73] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[74] Amol Deshpande and Lisa Hellerstein. Flow Algorithms for Parallel Query Optimization. In *Proc. of the 24th IEEE Intl. Conf. on Data Engineering*, pages 754–763. IEEE, 2008.

[75] David J DeWitt, Shahram Ghandeharizadeh, Donovan A. Schneider, Allan Bricker, H-I Hsiao, and Rick Rasmussen. The Gamma Database Machine Project. *IEEE Trans. on Knowledge and Data Engineering*, 2(1):44–62, 1990.

[76] David J. DeWitt and Jim Gray. Parallel Database Systems: The Future of High Performance Database Systems. *Communications of the ACM*, 35(6):85–98, 1992.

[77] David J. DeWitt, Jeffrey F. Naughton, Donovan A. Schneider, and S. Seshadri. Practical Skew Handling in Parallel Joins. In *Proc. of the 18th Intl. Conf. on Very Large Data Bases*, pages 27–40. VLDB Endowment, 1992.

[78] Karl Dias, Mark Ramacher, Uri Shaft, Venkateshwaran Venkataramani, and Graham Wood. Automatic Performance Diagnosis and Tuning in Oracle. In *Proc. of the 3rd Biennial Conf. on Innovative Data Systems Research*, pages 84–94, 2005.

[79] Jens Dittrich and Jorge-Arnulfo Quiané-Ruiz. Efficient Big Data Processing in Hadoop MapReduce. *Proc. of the VLDB Endowment*, 5(12):2014–2015, 2012.

[80] Jens Dittrich, Jorge-Arnulfo Quiané-Ruiz, Alekh Jindal, Yagiz Kargin, Vinay Setty, and Jörg Schad. Hadoop++: Making a Yellow Elephant Run Like a Cheetah. *Proc. of the VLDB Endowment*, 3(1-2):515–529, 2010.

[81] Jens Dittrich, Jorge-Arnulfo Quiané-Ruiz, Stefan Richter, Stefan Schuh, Alekh Jindal, and Jörg Schad. Only Aggressive Elephants are Fast Elephants. *Proc. of the VLDB Endowment*, 5(11):1591–1602, 2012.

[82] *3-D Data Management: Controlling Data Volume, Velocity and Variety*, 2013. Doug Laney, Research Note, META Group, February 2001.

[83] Jaliya Ekanayake, Hui Li, Bingjing Zhang, Thilina Gunarathne, Seung-Hee Bae, Judy Qiu, and Geoffrey Fox. Twister: A Runtime for Iterative MapReduce. In *Proc. of the 19th Intl. Symposium on High Performance Distributed Computing*, pages 810–818. ACM, 2010.

[84] Mohamed Y Eltabakh, Yuanyuan Tian, Fatma Özcan, Rainer Gemulla, Aljoscha Krettek, and John McPherson. CoHadoop: Flexible Data Placement and its Exploitation in Hadoop. *Proc. of the VLDB Endowment*, 4(9):575–585, 2011.

[85] Susanne Englert, Jim Gray, Terrye Kocher, and Praful Shah. A Benchmark of NonStop SQL Release 2 Demonstrating Near-Linear Speedup and Scaleup on Large Databases. In *Proc. of the 1990 ACM SIGMETRICS Intl. Conf. on Measurement and Modeling of Computer Systems*, pages 245–246. ACM, 1990.

[86] *Esper*, 2013. `http://esper.codehaus.org/`.

[87] Stephan Ewen, Kostas Tzoumas, Moritz Kaufmann, and Volker Markl. Spinning Fast Iterative Data Flows. *Proc. of the VLDB Endowment*, 5(11):1268–1279, 2012.

[88] Franz Färber, Norman May, Wolfgang Lehner, Philipp Große, Ingo Müller, Hannes Rauhe, and Jonathan Dees. The SAP HANA Database – An Architecture Overview. *IEEE Data Engineering Bulletin*, 35(1):28–33, 2012.

[89] Avrilia Floratou, Jignesh M Patel, Eugene J Shekita, and Sandeep Tata. Column-oriented Storage Techniques for MapReduce. *Proc. of the VLDB Endowment*, 4(7):419–429, 2011.

[90] Avrilia Floratou, Nikhil Teletia, David J DeWitt, Jignesh M Patel, and Donghui Zhang. Can the Elephants Handle the NoSQL Onslaught? *Proc. of the VLDB Endowment*, 5(12):1712–1723, 2012.

[91] Michael J Franklin, Sailesh Krishnamurthy, Neil Conway, Alan Li, Alex Russakovsky, and Neil Thombre. Continuous Analytics: Rethinking Query Processing in a Network-Effect World. In *Proc. of the 4th Biennial Conf. on Innovative Data Systems Research*. Citeseer, 2009.

[92] Eric Friedman, Peter Pawlowski, and John Cieslewicz. SQL/MapReduce: A Practical Approach to Self-Describing, Polymorphic, and Parallelizable User-Defined Functions. *Proc. of the VLDB Endowment*, 2(2):1402–1413, 2009.

[93] Sumit Ganguly, Waqar Hasan, and Ravi Krishnamurthy. Query Optimization for Parallel Execution. In *Proc. of the 1992 ACM SIGMOD Intl. Conf. on Management of Data*, pages 9–18. ACM, 1992.

[94] Alan F. Gates, Olga Natkovich, Shubham Chopra, Pradeep Kamath, Shravan M. Narayanamurthy, Christopher Olston, Benjamin Reed, Santhosh Srinivasan, and Utkarsh Srivastava. Building a High-Level Dataflow System on top of Map-Reduce: The Pig Experience. *Proc. of the VLDB Endowment*, 2(2):1414–1425, 2009.

[95] Lars George. *HBase: The Definitive Guide*. O'Reilly Media, 2011.

[96] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The Google File System. *ACM SIGOPS Operating Systems Review*, 37(5):29–43, 2003.

[97] Goetz Graefe. Volcano - An Extensible and Parallel Query Evaluation System. *IEEE Trans. on Knowledge and Data Engineering*, 6(1):120–135, 1994.

[98] Goetz Graefe. The Cascades Framework for Query Optimization. *IEEE Data Engineering Bulletin*, 18(3):19–29, 1995.

[99] *Greenplum*, 2012. `http://www.greenplum.com`.

[100] Raman Grover and Michael J. Carey. Extending Map-Reduce for Efficient Predicate-Based Sampling. In *Proc. of the 28th IEEE Intl. Conf. on Data Engineering*, pages 486–497. IEEE, 2012.

[101] Martin Grund, Philippe Cudré-Mauroux, Jens Krüger, Samuel Madden, and Hasso Plattner. An overview of HYRISE - a Main Memory Hybrid Storage Engine. *IEEE Data Engineering Bulletin*, 35(1):52–57, 2012.

[102] Alexander Hall, Olaf Bachmann, Robert Büssow, Silviu Gănceanu, and Marc Nunkesser. Processing a Trillion Cells per Mouse Click. *Proc. of the VLDB Endowment*, 5(11):1436–1446, 2012.

[103] Wook-Shin Han, Jack Ng, Volker Markl, Holger Kache, and Mokhtar Kandil. Progressive Optimization in a Shared-nothing Parallel Database. In *Proc. of the 2007 ACM SIGMOD Intl. Conf. on Management of Data*, pages 809–820. ACM, 2007.

[104] Waqar Hasan and Rajeev Motwani. Coloring Away Communication in Parallel Query Optimization. In *Proc. of the 21st Intl. Conf. on Very Large Data Bases*, pages 239–250. VLDB Endowment, 1995.

[105] Yongqiang He, Rubao Lee, Yin Huai, Zheng Shao, Namit Jain, Xiaodong Zhang, and Zhiwei Xu. RCFile: A Fast and Space-efficient Data Placement Structure in MapReduce-based Warehouse Systems. In *Proc. of the 27th IEEE Intl. Conf. on Data Engineering*, pages 1199–1208. IEEE, 2011.

[106] Joseph M Hellerstein, Peter J Haas, and Helen J Wang. Online Aggregation. In *Proc. of the 1997 ACM SIGMOD Intl. Conf. on Management of Data*, pages 171–182. ACM, 1997.

[107] Herodotos Herodotou and Shivnath Babu. Xplus: A SQL-Tuning-Aware Query Optimizer. *Proc. of the VLDB Endowment*, 3(1-2):1149–1160, 2010.

[108] Herodotos Herodotou, Nedyalko Borisov, and Shivnath Babu. Query Optimization Techniques for Partitioned Tables. In *Proc. of the 2011 ACM SIGMOD Intl. Conf. on Management of Data*, pages 49–60. ACM, 2011.

[109] Herodotos Herodotou, Fei Dong, and Shivnath Babu. No One (Cluster) Size Fits All: Automatic Cluster Sizing for Data-intensive Analytics. In *Proc. of the 2nd Symp. on Cloud Computing*. ACM, 2011.

[110] Herodotos Herodotou, Harold Lim, Gang Luo, Nedyalko Borisov, Liang Dong, Fatma Bilgen Cetin, and Shivnath Babu. Starfish: A Self-tuning System for Big Data Analytics. In *Proc. of the 5th Biennial Conf. on Innovative Data Systems Research*, 2011.

[111] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D Joseph, Randy Katz, Scott Shenker, and Ion Stoica. Mesos: A Platform for Fine-grained Resource Sharing in the Data Center. In *Proc. of the 8th USENIX Symp. on Networked Systems Design and Implementation*. USENIX Association, 2011.

[112] Jeffrey A. Hoffer and Dennis G. Severance. The Use of Cluster Analysis in Physical Data Base Design. In *Proc. of the 1st Intl. Conf. on Very Large Data Bases*, pages 69–86. ACM, 1975.

[113] Wei Hong and Michael Stonebraker. Optimization of Parallel Query Execution Plans in XPRS. *Distributed and Parallel Databases*, 1(1):9–32, 1993.

[114] Hui-I Hsiao and David J. DeWitt. Chained Declustering: A New Availability Strategy for Multiprocessor Database Machines. In *Proc. of the 6th IEEE Intl. Conf. on Data Engineering*, pages 456–465, 1990.

[115] IBM Corporation2. *Partitioned Tables*, 2007. `http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.admin.partition.doc/doc/c0021560.html`.

[116] *IBM Netezza Data Warehouse Appliances*, 2012. `http://www-01.ibm.com/software/data/netezza/`.

[117] Stratos Idreos, Fabian Groffen, Niels Nes, Stefan Manegold, K. Sjoerd Mullender, and Martin L. Kersten. MonetDB: Two Decades of Research in Column-oriented Database Architectures. *IEEE Data Engineering Bulletin*, 35(1):40–45, 2012.

[118] *Infobright*, 2013. `http://www.infobright.com/`.

[119] Michael Isard, Mihai Budiu, Yuan Yu, Andrew Birrell, and Dennis Fetterly. Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks. *ACM SIGOPS Operating Systems Review*, 41(3):59–72, 2007.

[120] Michael Isard and Yuan Yu. Distributed Data-Parallel Computing Using a High-Level Programming Language. In *Proc. of the 2009 ACM SIGMOD Intl. Conf. on Management of Data*, pages 987–994. ACM, 2009.

[121] Ming-Yee Iu and Willy Zwaenepoel. HadoopToSQL: A MapReduce Query Optimizer. In *Proc. of the 5th European Conf. on Computer Systems*, pages 251–264. ACM, 2010.

[122] Alekh Jindal, Jorge-Arnulfo Quian-Ruiz, and Jens Dittrich. Trojan Data Layouts: Right Shoes for a Running Elephant. In *Proc. of the 2nd Symp. on Cloud Computing*. ACM, 2011.

[123] Tim Kaldewey, Eugene J Shekita, and Sandeep Tata. Clydesdale: Structured Data Processing on MapReduce. In *Proc. of the 15th Intl. Conf. on Extending Database Technology*, pages 15–25. ACM, 2012.

[124] Kamal Kc and Kemafor Anyanwu. Scheduling Hadoop Jobs to Meet Deadlines. In *Proc. of the 2nd IEEE Intl. Conf. on Cloud Computing Technology and Science*, pages 388–392. IEEE, 2010.

[125] Alfons Kemper, Thomas Neumann, Florian Funke, Viktor Leis, and Henrik Mühe. HyPer: Adapting Columnar Main-Memory Data Management for Transactional AND Query Processing. *IEEE Data Engineering Bulletin*, 35(1):46–51, 2012.

[126] Martin L. Kersten, Ying Zhang, Milena Ivanova, and Niels Nes. SciQL, a Query Language for Science Applications. In *Proc. of the EDBT/ICDT Workshop on Array Databases*, pages 1–12. ACM, 2011.

[127] *Kosmos Distributed Filesystem*, 2013. `http://code.google.com/p/kosmosfs/`.

[128] Donald Kossmann. The State of the Art in Distributed Query Processing. *ACM Computing Surveys (CSUR)*, 32(4):422–469, 2000.

[129] Stefan Krompass, Umeshwar Dayal, Harumi A. Kuno, and Alfons Kemper. Dynamic Workload Management for Very Large Data Warehouses: Juggling Feathers and Bowling Balls. In *Proc. of the 33rd Intl. Conf. on Very Large Data Bases*, pages 1105–1115. VLDB Endowment, 2007.

[130] Stefan Krompass, Harumi A. Kuno, Janet L. Wiener, Kevin Wilkinson, Umeshwar Dayal, and Alfons Kemper. Managing Long-running Queries. In *Proc. of the 13th Intl. Conf. on Extending Database Technology*, pages 132–143. ACM, 2009.

[131] Avinash Lakshman and Prashant Malik. Cassandra: A Decentralized Structured Storage System. *Operating Systems Review*, 44(2):35–40, 2010.

[132] Wang Lam, Lu Liu, STS Prasad, Anand Rajaraman, Zoheb Vacheri, and AnHai Doan. Muppet: MapReduce-Style Processing of Fast Data. *Proc. of the VLDB Endowment*, 5(12):1814–1825, 2012.

[133] Andrew Lamb, Matt Fuller, Ramakrishna Varadarajan, Nga Tran, Ben Vandiver, Lyric Doshi, and Chuck Bear. The Vertica Analytic Database: C-store 7 Years Later. *Proc. of the VLDB Endowment*, 5(12):1790–1801, 2012.

[134] Rosana S. G. Lanzelotte, Patrick Valduriez, and Mohamed Zaït. On the Effectiveness of Optimization Search Strategies for Parallel Execution Spaces. In *Proc. of the 19th Intl. Conf. on Very Large Data Bases*, pages 493–504. Morgan Kaufmann Publishers Inc., 1993.

[135] Nikolay Laptev, Kai Zeng, and Carlo Zaniolo. Early Accurate Results for Advanced Analytics on MapReduce. *Proc. of the VLDB Endowment*, 5(10):1028–1039, 2012.

[136] Kyong-Ha Lee, Yoon-Joon Lee, Hyunsik Choi, Yon Dohn Chung, and Bongki Moon. Parallel Data Processing with MapReduce: a Survey. *ACM SIGMOD Record*, 40(4):11–20, 2011.

[137] Rubao Lee, Tian Luo, Yin Huai, Fusheng Wang, Yongqiang He, and Xiaodong Zhang. YSmart: Yet Another SQL-to-MapReduce Translator. In *Proc. of the 31st Intl. Conf. on Distributed Computing Systems*, pages 25–36. IEEE, 2011.

[138] Marcus Leich, Jochen Adamek, Moritz Schubotz, Arvid Heise, Astrid Rheinländer, and Volker Markl. Applying Stratosphere for Big Data Analytics. In *Proc. of the 15th USENIX Annual Technical Conference*, pages 507–510, 2013.

[139] Harold Lim, Herodotos Herodotou, and Shivnath Babu. Stubby: A Transformation-based Optimizer for MapReduce Workflows. *Proc. of the VLDB Endowment*, 5(11):1196–1207, 2012.

[140] Yuting Lin, Divyakant Agrawal, Chun Chen, Beng Chin Ooi, and Sai Wu. Llama: Leveraging Columnar Storage for Scalable Join Processing in the MapReduce Framework. In *Proc. of the 2011 ACM SIGMOD Intl. Conf. on Management of Data*, pages 961–972. ACM, 2011.

[141] Yucheng Low, Joseph Gonzalez, Aapo Kyrola, Danny Bickson, Carlos Guestrin, and Joseph M. Hellerstein. Distributed GraphLab: A Framework for Machine Learning in the Cloud. *Proc. of the VLDB Endowment*, 5(8):716–727, 2012.

[142] Hongjun Lu. *Query Processing in Parallel Relational Database Systems.* IEEE Computer Society Press, 1st edition, 1994.

[143] Hongjun Lu, Ming-Chien Shan, and Kian-Lee Tan. Optimization of Multi-Way Join Queries for Parallel Execution. In *Proc. of the 17th Intl. Conf. on Very Large Data Bases*, pages 549–560. VLDB Endowment, 1991.

[144] Hongjun Lu and Kian-Lee Tan. Dynamic and Load-balanced Task-Oriented Datbase Query Processing in Parallel Systems. In *Proc. of the 3rd Intl. Conf. on Extending Database Technology*, pages 357–372. ACM, 1992.

[145] Hongjun Lu and Kian-Lee Tan. Load Balanced Join Processing in Shared-Nothing Systems. *Journal of Parallel and Distributed Computing*, 23(3):382–398, 1994.

[146] Soren Macbeth. *Why YieldBot Chose Cascalog over Pig for Hadoop Processing*, 2011. `http://tech.backtype.com/52456836`.

[147] Roger MacNicol and Blaine French. Sybase IQ Multiplex-designed for Analytics. In *Proc. of the 30th Intl. Conf. on Very Large Data Bases*, pages 1227–1230. VLDB Endowment, 2004.

[148] Grzegorz Malewicz, Matthew H Austern, Aart JC Bik, James C Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. Pregel: A System for Large-scale Graph Processing. In *Proc. of the 2010 ACM SIGMOD Intl. Conf. on Management of Data*, pages 135–146. ACM, 2010.

[149] *MapR File System*, 2013. `http://www.mapr.com/products/apache-hadoop`.

[150] Frank McSherry, Derek Gordon Murray, Rebecca Isaacs, and Michael Isard. Differential Dataflow. In *Proc. of the 6th Biennial Conf. on Innovative Data Systems Research*, 2013.

[151] Manish Mehta and David J. DeWitt. Data Placement in Shared-Nothing Parallel Database Systems. *The VLDB Journal*, 6(1):53–72, 1997.

[152] Erik Meijer, Brian Beckman, and Gavin Bierman. LINQ: Reconciling Object, Relations and XML in the .NET Framework. In *Proc. of the 2006 ACM SIGMOD Intl. Conf. on Management of Data*, pages 706–706. ACM, 2006.

[153] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, and Theo Vassilakis. Dremel: Interactive Analysis of Web-Scale Datasets. *Proc. of the VLDB Endowment*, 3(1):330–339, 2010.

[154] Svilen R Mihaylov, Zachary G Ives, and Sudipto Guha. REX: Recursive, Delta-based Data-centric Computation. *Proc. of the VLDB Endowment*, 5(11):1280–1291, 2012.

[155] Tony Morales. *Oracle Database VLDB and Partitioning Guide 11g Release 1 (11.1)*. Oracle Corporation, 2007. `http://docs.oracle.com/cd/B28359_01/server.111/b32024.pdf`.

[156] Leonardo Neumeyer, Bruce Robbins, Anish Nair, and Anand Kesari. S4: Distributed Stream Computing Platform. In *Proc. of the 2010 IEEE Intl. Conf. on Data Mining Workshops*. IEEE, 2010.

[157] Tomasz Nykiel, Michalis Potamias, Chaitanya Mishra, George Kollios, and Nick Koudas. MRShare: Sharing Across Multiple Queries in MapReduce. *Proc. of the VLDB Endowment*, 3(1):494–505, 2010.

[158] Christopher Olston, Greg Chiou, Laukik Chitnis, Francis Liu, Yiping Han, Mattias Larsson, Andreas Neumann, Vellanki B. N. Rao, Vijayanand Sankarasubramanian, Siddharth Seth, Chao Tian, Topher ZiCornell, and Xiaodan Wang. Nova: Continuous Pig/Hadoop Workflows. In *Proc. of the 2011 ACM SIGMOD Intl. Conf. on Management of Data*, pages 1081–1090. ACM, 2011.

[159] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins. Pig Latin: A Not-So-Foreign Language for Data Processing. In *Proc. of the 2008 ACM SIGMOD Intl. Conf. on Management of Data*, pages 1099–1110. ACM, 2008.

[160] Patrick E. O'Neil, Edward Cheng, Dieter Gawlick, and Elizabeth J. O'Neil. The Log-Structured Merge-Tree (LSM-Tree). *Acta Informatica*, 33(4):351–385, 1996.

[161] *Oozie: Workflow Engine for Hadoop*, 2010. `http://yahoo.github.com/oozie/`.

[162] Michael Ovsiannikov, Silvius Rus, Damian Reeves, Paul Sutter, Sriram Rao, and Jim Kelly. The Quantcast File System. *Proc. of the VLDB Endowment*, 6(11), 2013.

[163] HweeHwa Pang, Michael J. Carey, and Miron Livny. Multiclass Query Scheduling in Real-Time Database Systems. *IEEE Trans. on Knowledge and Data Engineering*, 7(4):533–551, 1995.

[164] *ParAccel Analytic Platform*, 2013. `http://www.paraccel.com/`.

[165] David A. Patterson, Garth A. Gibson, and Randy H. Katz. A Case for Redundant Arrays of Inexpensive Disks (RAID). In *Proc. of the 1985 ACM SIGMOD Intl. Conf. on Management of Data*, pages 109–116, 1988.

[166] Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel Abadi, David DeWitt, Samuel Madden, and Michael Stonebraker. A Comparison of Approaches to Large-Scale Data Analysis. In *Proc. of the 2009 ACM SIGMOD Intl. Conf. on Management of Data*, pages 165–178. ACM, 2009.

[167] *Protocol Buffers Developer Guide*, 2012. `https://developers.google.com/protocol-buffers/docs/overview`.

[168] B Thirumala Rao and LSS Reddy. Survey on Improved Scheduling in Hadoop MapReduce in Cloud Environments. *Intl. Journal of Computer Applications*, 34(9):29–33, 2011.

[169] Jun Rao, Chun Zhang, Nimrod Megiddo, and Guy M. Lohman. Automating Physical Database Design in a Parallel Database. In *Proc. of the 2002 ACM SIGMOD Intl. Conf. on Management of Data*, pages 558–569, 2002.

[170] Sriram Rao, Raghu Ramakrishnan, Adam Silberstein, Mike Ovsiannikov, and Damian Reeves. Sailfish: A Framework for Large Scale Data Processing. In *Proc. of the 3rd Symp. on Cloud Computing*, page 4. ACM, 2012.

[171] Joshua Rosen, Neoklis Polyzotis, Vinayak R. Borkar, Yingyi Bu, Michael J. Carey, Markus Weimer, Tyson Condie, and Raghu Ramakrishnan. Iterative MapReduce for Large Scale Machine Learning. *Computing Research Repository (CoRR)*, abs/1303.3517, 2013.

[172] Thomas Sandholm and Kevin Lai. Dynamic Proportional Share Scheduling in Hadoop. In *Proc. of the 15th IEEE Intl. Conf. on Data Mining*, pages 110–131. Springer, 2010.

[173] Patricia G. Selinger, Morton M Astrahan, Donald D. Chamberlin, Raymond A Lorie, and Thomas G. Price. Access Path Selection in a Relational Database Management System. In *Proc. of the 1979 ACM SIGMOD Intl. Conf. on Management of Data*, pages 23–34. ACM, 1979.

[174] Kyuseok Shim. MapReduce Algorithms for Big Data Analysis. *Proc. of the VLDB Endowment*, 5(12):2016–2017, 2012.

[175] Avraham Shinnar, David Cunningham, Vijay Saraswat, and Benjamin Herta. M3R: Increased Performance for In-memory Hadoop Jobs. *Proc. of the VLDB Endowment*, 5(12):1736–1747, 2012.

[176] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The Hadoop Distributed File System. In *Proc. of the 26th IEEE Symp. on Mass Storage Systems and Technologies*, pages 1–10. IEEE, 2010.

[177] *SQL Server Parallel Data Warehouse*, 2012. `http://www.microsoft.com/en-us/sqlserver/solutions-technologies/data-warehousing/pdw.aspx`.

[178] Garrick Staples. TORQUE Resource Manager. In *Proc. of the 20th ACM Intl. Conf. on Supercomputing*, page 8. ACM, 2006.

[179] Michael Stillger, Myra Spiliopoulou, and Johann Christoph Freytag. *Parallel Query Optimization: Exploiting Bushy and Pipeline Parallelisms with Genetic Programs.* Citeseer, 1996.

[180] Michael Stonebraker, Paul Brown, Alex Poliakov, and Suchi Raman. The Architecture of SciDB. In *Proc. of the 23rd Intl. Conf. on Scientific and Statistical Database Management*, pages 1–16. IEEE, 2011.

[181] Mike Stonebraker, Daniel J Abadi, Adam Batkin, Xuedong Chen, Mitch Cherniack, Miguel Ferreira, Edmond Lau, Amerson Lin, Sam Madden, Elizabeth O'Neil, et al. C-Store: A Column-oriented DBMS. In *Proc. of the 31st Intl. Conf. on Very Large Data Bases*, pages 553–564. VLDB Endowment, 2005.

[182] *Storm*, 2013. `http://storm-project.net/`.

[183] *StreamBase*, 2013. `http://www.streambase.com/`.

[184] Michal Switakowski, Peter A. Boncz, and Marcin Zukowski. From Cooperative Scans to Predictive Buffer Management. *Proc. of the VLDB Endowment*, 5(12):1759–1770, 2012.

[185] Sybase, Inc. *Performance and Tuning: Optimizer and Abstract Plans*, 2003. `http://infocenter.sybase.com/help/topic/com.sybase.dc20023_1251/pdf/optimizer.pdf`.

[186] Ron Talmage. *Partitioned Table and Index Strategies Using SQL Server 2008.* Microsoft, 2009. `http://msdn.microsoft.com/en-us/library/dd578580.aspx`.

[187] Kian-Lee Tan and Hongjun Lu. On Resource Scheduling of Multi-Join Queries. *Information Processing Letters*, 48(4):189–195, 1993.

[188] *Teradata*, 2012. `http://www.teradata.com`.

[189] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. Hive: A Warehousing Solution over a Map-Reduce Framework. *Proc. of the VLDB Endowment*, 2(2):1626–1629, 2009.

[190] Leslie G Valiant. A Bridging Model for Parallel Computation. *Communications of the ACM*, 33(8):103–111, 1990.

[191] Sage A Weil, Scott A Brandt, Ethan L Miller, Darrell DE Long, and Carlos Maltzahn. Ceph: A Scalable, High-performance Distributed File System. In *Proc. of the 7th USENIX Symp. on Operating Systems Design and Implementation*, pages 307–320. USENIX Association, 2006.

[192] Tom White. *Hadoop: The Definitive Guide*. Yahoo! Press, 2010.

[193] Sai Wu, Feng Li, Sharad Mehrotra, and Beng Chin Ooi. Query Optimization for Massively Parallel Data Processing. In *Proc. of the 2nd Symp. on Cloud Computing*. ACM, 2011.

[194] Reynold Xin, Joseph Gonzalez, and Michael Franklin. GraphX: A Resilient Distributed Graph System on Spark. In *Proc. of the ACM SIGMOD GRADES Workshop*. ACM, 2013.

[195] Reynold Xin, Josh Rosen, Matei Zaharia, Michael J Franklin, Scott Shenker, and Ion Stoica. Shark: SQL and Rich Analytics at Scale. Technical Report UCB/EECS-2012-214, University of California, Berkeley, 2012.

[196] Mark Yong, Nitin Garegrat, and Shiwali Mohan. Towards a Resource Aware Scheduler in Hadoop. In *Proc. of the 2009 IEEE Intl. Conf. on Web Services*, pages 102–109. IEEE, 2009.

[197] Yuan Yu, Pradeep Kumar Gunda, and Michael Isard. Distributed Aggregation for Data-parallel Computing: Interfaces and Implementations. In *Proc. of the 22nd ACM Symp. on Operating Systems Principles*, pages 247–260. ACM, 2009.

[198] Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma, Khaled Elmeleegy, Scott Shenker, and Ion Stoica. Delay Scheduling: A Simple Technique for Achieving Locality and Fairness in Cluster Scheduling. In *Proc. of the 5th European Conf. on Computer Systems*, pages 265–278. ACM, 2010.

[199] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael Franklin, Scott Shenker, and Ion Stoica. Resilient Distributed Datasets: A Fault-tolerant Abstraction for In-memory Cluster Computing. In *Proc. of the 9th USENIX Symp. on Networked Systems Design and Implementation*, pages 2–2. USENIX Association, 2012.

[200] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster Computing with Working Sets. In *Proc. of the 2nd USENIX Conf. on Hot Topics in Cloud Computing*. USENIX Association, 2010.

[201] Bernhard Zeller and Alfons Kemper. Experience Report: Exploiting Advanced Database Optimization Features for Large-Scale SAP R/3 Installations. In *Proc. of the 28st Intl. Conf. on Very Large Data Bases*, pages 894–905. VLDB Endowment, 2002.

[202] Yanfeng Zhang, Qixin Gao, Lixin Gao, and Cuirong Wang. PrIter: A Distributed Framework for Prioritized Iterative Computations. In *Proc. of the 2nd Symp. on Cloud Computing*, pages 13:1–13:14. ACM, 2011.

[203] Yi Zhang, Herodotos Herodotou, and Jun Yang. RIOT: I/O-Efficient Numerical Computing without SQL. In *Proc. of the 4th Biennial Conf. on Innovative Data Systems Research*, 2009.

[204] Jingren Zhou, Nicolas Bruno, Ming-Chuan Wu, Per-Ake Larson, Ronnie Chaiken, and Darren Shakib. SCOPE: Parallel Databases Meet MapReduce. *The VLDB Journal*, 21(5):611–636, 2012.

[205] Jingren Zhou, Per-Åke Larson, and Ronnie Chaiken. Incorporating Partitioning and Parallel Plans into the SCOPE Optimizer. In *Proc. of the 26th IEEE Intl. Conf. on Data Engineering*, pages 1060–1071. IEEE, 2010.

[206] M Zukowski and P Boncz. VectorWise: Beyond Column Stores. *IEEE Data Engineering Bulletin*, 35(1):21–27, 2012.

[207] Marcin Zukowski, Sándor Héman, Niels Nes, and Peter A. Boncz. Cooperative Scans: Dynamic Bandwidth Sharing in a DBMS. In *Proc. of the 33rd Intl. Conf. on Very Large Data Bases*, pages 723–734. VLDB Endowment, 2007.