

A Short Survey on the State of the Art in Architectures and Platforms for Large Scale Data Analysis and Knowledge Discovery from Data

Edmon Begoli
Oak Ridge National Laboratory
1 Bethel Valley Rd.
Oak Ridge, TN, USA
begolie@ornl.gov

ABSTRACT

Intended as a survey for practicing architects and researchers seeking an overview of the state-of-the-art architectures for data analysis, this paper provides an overview of the emerging data management and analytic platforms including parallel databases, Hadoop-based systems, High Performance Computing (HPC) platforms and platforms popularly referred to as NoSQL platforms. Platforms are presented based on their relevance, analysis they support and the data organization model they support.

Categories and Subject Descriptors

H.4 [Information Systems Applications]: Miscellaneous;
D.2.8 [Software Architecture]: Design—*parallel distributed architectures*

General Terms

Design

Keywords

Software Architecture, Large Scale Data Analysis, Knowledge Discovery from Data, Big Data, NoSQL, Massively Parallel Processing

1. INTRODUCTION

This paper is organized into four main sections - review of the terminology that will be frequently used in the paper when describing the architectures and trends in large scale data analysis, survey of the most popular knowledge discovery platforms and tools, taxonomic review of the architectural styles for data analysis and review of the most popular analytic platforms. Field of data analytic architectures is broad, rapidly changing and complex. We therefore attempt to survey the most relevant aspects of the field by reviewing it from four different aspects - terminology, usage surveys, taxonomies and implementations.

2. RELEVANT TERMINOLOGY

Before discussing the state-of-the-art architectures for knowledge discovery from data we review the key terminology and definitions used in the field of data analysis and knowledge discovery.

2.0.1 Knowledge Discovery from Data

Knowledge Discovery from Data (KDD) is a set of activities designed to extract new knowledge from complex and large datasets. It is an interdisciplinary field spanning computer science, statistics, cognitive science, visualization, and domain specific methods and expertise (e.g. medicine, economics, government).

2.0.2 Data Analysis

Given the broad scope of the field data analysis, in this paper we define data analysis as collection of the computational, statistical and visualization methods for better understanding of the data. Specifically, we consider these disciplines as integral to data analysis and for understanding of the requirements that drive the characteristics of the analytic architectures:

- Machine Learning
Machine learning is a discipline that encompasses statistical, probabilistic and data mining techniques for automated, repeatable and learnable classification, filtering and categorization of data. Machine learning techniques are broadly categorized as supervised, semi-supervised or unsupervised.
- Data Mining
Although term data mining term is often used interchangeably with Knowledge Discovery from Data (KDD), in the context of modern architectural practices where KDD often encompasses data related but not exclusively data focused operations, we define data mining more narrowly as discipline concerned with data organization, preparation and (automated) discovery of useful models and patterns in large datasets.
- Statistical Analysis
Statistical analysis is a collection of mathematical methods for exploratory and confirmatory analysis of large amounts of data and discovery of the relationships, trends and overall meaning of the data.

- Information Visualization and Visual Analysis

InfoVis[7] web site defines Information Visualization as "process of transforming information into a visual form enabling the viewer to observe, browse, make sense, and understand the information. It typically employs computers to process the information and computer screens to view it using methods of interactive graphics, imaging, and visual design. It relies on the visual system to perceive and process the information." Visual analysis is cognitively intensive but essentially computer-mediated activity.

2.0.3 Major Data Analysis Architectures

The remainder of the section defines key technological terminology and architectural concepts relevant for the survey of the state-of-the-art analytic data architectures.

- Data Warehouse

According to one of the fathers of the field, Dr. Ralph Kimball[23], data warehouse is "conglomeration of an organization's data warehouse staging and presentation areas, where operational data is specifically structured for query and analysis performance and ease-of-use".

- Massively Parallel Processing (MPP) Databases

MPP is a computational technique that employs large number of computational units (cores, processors of separate machines) to perform a set of coordinated computations in parallel. In the context of databases, MPP architectures store data distributively and process it in parallel.

- Shared-nothing Data Architecture

Shared-nothing term applies to the distributed data management systems. Each data management server manages and stores own copy of the data on its local disk storage. This locality of the data allows for faster data access since there is no network access overhead associated with the shared storage devices.

- Distributed Parallel File Systems

Distributed file systems are the foundational and fundamental mechanisms for large scale data processing. Distributed parallel file systems stripe data over multiple servers for high performance, parallel data access and computation.

- Hadoop

Hadoop is an open source, Java-based programming framework that supports the processing of large data sets in a distributed computing environment. Design of Hadoop's core components, HDFS and MapReduce was inspired by Google's Google File System [19] and MapReduce frameworks [16].

- Key-value stores

Key-values stores[29] are schema-less data management systems with simplistic, key-value based data manipulation semantics - inserts, updates and deletes.

- NoSQL

NoSQL is an umbrella term for the family of data management technologies characterized by no or simple schema organization, key-value semantics and highly scalable "share nothing" storage organization. There is no single definition that completely defines this popular concept. It is generally accepted that NoSQL databases do not offer SQL semantics; they are not ACID compliant, and they have distributed, fault-tolerant storage architecture. There are however exceptions to any of these characteristics within universe of technologies considered to be NoSQL databases.

- Hadoop File System DFS

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. It is not a file system in an operating systems sense - it is data storage and management layer implemented in Java that features built-in, high fault-tolerance. [13]

- MapReduce

MapReduce is a divide-and-conquer algorithm for parallel processing of data. Its first step, *map*, maps the function onto a key-indexed data chunks. Upon the application of the function, results are *reduced*, using the same key-indexed scheme into a result set.

- Column oriented data stores

Traditional relational databases store its data in row oriented fashion i.e. fields a_i and a_j of the row a are stored contiguously and they precede all the fields of the row b . In column oriented databases (aka "Columnars"), data is stored by the columns - i.e. fields a_1 , b_1 , c_1 of the rows a, b, c are stored contiguously and usually on a separate data node in a shared-nothing architecture. Column orientation offers performance benefits for the data analytic operations where tabularly organized data is often accessed, aggregated and operated on along the columns in a table.

- Column compression

Related to column orientation, column compressing databases can store the compressed version of their columns by storing only counts of the redundant data instead of redundant data itself (for example, store count of 10,000 repeated digits 0, instead of storing actual number zero 10,000 times).

3. KDD PLATFORM ADOPTION STATISTICS

We begin the survey of the architectures by presenting the recent survey[22] conducted by KDnuggets web site. Survey data shows the current state of adoption of data mining software, big data technologies and programming languages used for data mining and knowledge discovery.

We present here an abbreviated list showing top ten most popular software packages. (Reference the KDnuggets web site for the full list of packages and reported use.)

Table 1: Most Popular Data Mining Software

Software	Respondents	2012	2011
R	245	30.7%	23.3%
Excel	238	29.8%	21.8%
Rapid-IRapid Miner	213	26.7%	27.7%
KNIME	174	21.8%	12.1%
Weka/Pentaho	118	14.8%	11.8%
StatSoft Statistica	112	14.0%	8.5%
SAS	101	12.7%	13.6%
Rapid-IRapid Analytics	83	10.4%	N/A
MATLAB	(80)	10.0%	7.2%
IBM SPSS Statistics	62	7.8%	7.2%

Figure 1: Data mining software

KDNuggets survey for 2012 also includes a new statistic on the adoption of "Big Data" technologies for data analysis and knowledge discovery:

Table 2: Big Data Technologies used in 2012

Software	Respondents	2012
Hadoop/Hbase/Pig/Hive	67	8.4%
Amazon Web Services (AWS)	36	4.5%
NoSQL databases	33	4.1%
Other Big Data Data Software	21	2.6%
Other Hadoop-based tools	10	1.3%

Figure 2: "Big Data" technologies

As data analysis is often highly specialized activity, development of the custom codes is customary. KDNuggets survey includes the statistics on use of programming languages in such tasks. Data indicates the dominance of the R and SQL programming languages, which, as reported, are being used by over 50% of the survey respondents.

Table 3: Programming Languages

Programming Language	Respondents	2012
R	245	30.7%
SQL	185	23.2%
Java	138	17.3%
Python	119	14.9%
C/C++	66	8.3%
Other languages	57	7.1%
Perl	37	4.6%
Awk/Gawk/Shell	31	3.9%
F#	5	0.6%

Figure 3: Programming languages used in custom codes for data mining

4. ARCHITECTURES

In this section we outline the most common platforms for knowledge discovery from data; we define their essential architectural properties, and we describe the representative, the most widely adopted implementations.

4.1 Relational and Data Warehouse Management Systems

The oldest and the most established platform of all presented, relational data warehouses have been both challenged and transformed by the exponential increase in needs for data collection, storage and analysis. Traditional, monolithic data warehouses have evolved into massively parallel processing (MPP) systems capable of processing petabytes of data[26].

Although MPP architectures vary (e.g. master-master or master-slave), in its most common implementation MPP databases consist of a master node and a multiple, shared-nothing, parallel slave segments connected via high speed network interconnect. These systems also offer either exclusively column orientation and column compression or hybrid column/row orientation with column compression.

Commercially, this is a very competitive market with leaders, according to Gartner's annual Data Warehouse market survey[12], being Teradata, Oracle, IBM, EMC Greenplum, SAP/Sybase and Microsoft.

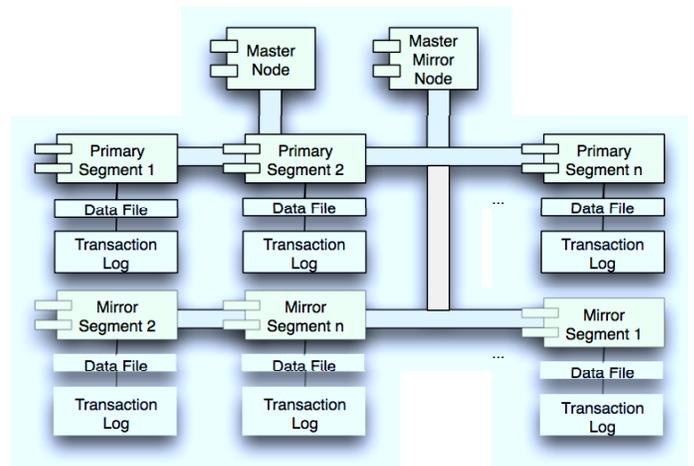


Figure 4: Massively Parallel Processing Database Architecture (Master-Slave)

4.2 Hadoop

Apache Hadoop[33] is an umbrella project for number of Java-based software packages and APIs for data management, organization, analysis and job scheduling at a very large scale. At the core of Hadoop are HDFS - Hadoop File System and MapReduce programming API. In addition to HDFS and MapReduce API, Hadoop consists of data warehousing software (Hive), massive scale column oriented database management software (HBase), machine learning library (Mahout), job tracking and scheduling suite (ZooKeeper and Pig) and other related libraries and packages for massively parallel and distributed data management.

Hadoop is used extensively in industry and academia for variety of tasks with log mining, structured and unstructured data analysis being some of the most frequent use cases.

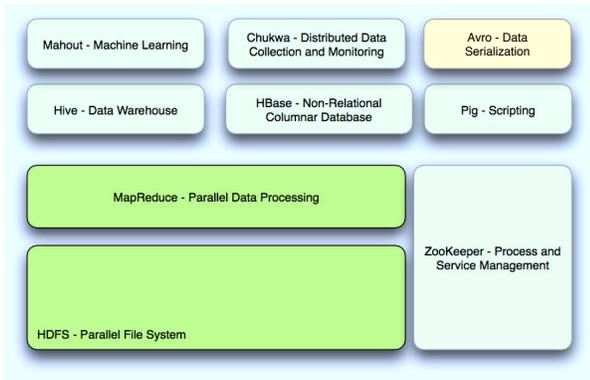


Figure 5: Hadoop Technology Stack

4.3 NoSQL

NoSQL databases in general offer flexible and scalable storage and query services under usually open source licenses. NoSQL databases are available with most of the public and private Cloud services offering an attractive solution for highly scalable, low cost data management needs.

Within NoSQL movement, dominant database architecture models are:

- Key value stores
- Document stores
- Graph databases

In this paper, within NoSQL domain, we give a somewhat more detailed overview of the Graph databases.

4.3.1 Graph databases

Graph databases are specialized data stores featuring semantics based on concepts from graph theory, and they use nodes, edges, edge labels and attributes as primary means for logical data representation. Primary value proposition of the graph databases is their ease of use and high performance of associative and random data access.

Physical implementation of graph databases differs, and it is not unlikely to find columnar, key-value or some other data stores as underlying persistence mechanisms. Primary use of graph databases is in flexible representation of asymmetric relationships and in mining of social networks.

Code sample below shows the ease of use in working with graph database, and how creating and saving associations on the fly dynamically augments schema-less representation.

```
from bulbs.neo4jserver import Graph,Config,NEO4J_URI

# connection logic
config = Config(NEO4J_URI, "username", "password")
```

```
g = Graph(config)
```

```
#lookup medical provider by its indexed attribute ID
providers = g.vertices.index.lookup(provider_id="101")
```

```
#iterate through the list of retrieved providers
for provider in providers:
    patients = provider.outV("treats")
```

```
# if this patient is in medicare then this provider
# serves patients in medicare and medicaid
# - it is dual provider
if patient.primary_plan == "medicare":
    provider.dual = True
    provider.save()
```

Using Neo4J and Pyhon Bulbs to flexibly query and add attributes on the fly

Some of the leading implementations of Graph databases are Neo4J,AllegroGraph,InfiniteGraph, HyperGraphDB, InfoGrid and Google Pregel.

4.4 Hadoop-Relational Hybrids

There is an active, ongoing discussion in database community of what is a proper delineation between Hadoop-based, "NoSQL" and relational derivatives[9], [32].

We outline here some of the representative implementations that offer different style of SQL-NoSQL amalgamation.

4.4.1 Google BigQuery

BigQuery is Google's commercial service based on the Dremel[25] platform for interactive analysis of very large scale data sets. Dremel combines ideas from web search and parallel DBMS architectures:

- Serving-tree based query processing - Dremel's query processing design borrows the concept of a serving tree found in distributed search engines [15]. In Dremel, query is percolated down the tree and rewritten at each step. The results are assembled by aggregating the replies received from lower levels of the tree.
- SQL-like semantics - Dremel provides a high-level, SQL-like language to express ad hoc queries. Dremel executes queries natively without translating them into MapReduce jobs.
- Column-oriented storage - Dremel uses a column-oriented storage representation.

4.4.2 HadoopDB

HadoopDB[10] is a hybrid implementation of the parallel DBMS and Hadoop. HadoopDB utilizes relational nodes for fast execution of queries and Hadoop job tracking and metadata catalog infrastructure for scheduling, tracking and results assembly of parallel queries.

HadoopDB consists of four components:

- Database Connector - The connector executes a SQL query on the database and returns results as key value pairs.
- Catalog - The catalog keeps the information needed to access the databases and metadata related to cluster data sets, replica locations and data partitions.
- Data loader - The data loader executes MapReduce job over Hadoop that reads the raw data files and partitions them into as many parts as the number of nodes in the cluster.
- SQL to MapReduce to SQL planner - The SQL-MR-SQL planner provides a parallel database front end for SQL queries. The planner translates the SQL queries into MapReduce jobs and optimizes the query plans for efficiency.

HadoopDB's reported performance approaches the performance and efficiency of parallel databases while offering scalability, fault tolerance and flexibility of Hadoop-based systems. HadoopDB is commercially offered as Hadapt[?].

4.4.3 Parallel Databases with Hadoop Connectors

This type of architecture is generally implemented as a massively parallel, RDBMS-based system capable of connecting into Hadoop (HDFS) for data loading and execution of MapReduce jobs. Most solutions in this space offer some form of native MapReduce-SQL semantics as well. Three most prominent representatives of this style of the architecture are Greenplum[3], Aster[?] and Vertica [4].

4.5 High Performance Computing(HPC) Platforms

High Performance Computing (HPC) is a discipline of a computational science focused on solving complex and computationally intensive science, engineering and business problems on specialized platforms. Computations performed on HPC platforms are usually characterized by massive parallelism and movement of "data to computation" which requires high bandwidth, low latency networking, massive number of compute cores (1000s) and high memory capacity as well as specialized software tools, APIs and algorithms. HPC platforms often incorporate General Processing Units (GPUs) for numerically intensive computing.

4.6 Others

In addition to parallel, Hadoop-based and graph oriented data analysis systems there are also families of data analytic architectures supporting specialized sub-domains of knowledge discovery: text/document and XML mining, geo-spatial, audio, video and signals analysis.

5. ARCHITECTURAL TAXONOMIES

In this section we survey some of the architectural taxonomies influenced by the most important constraints in data analysis - completion time bounds for analysis and data volume, representation and storage considerations.

5.1 Temporally Characterized Architectures

Temporal bounds for data analysis have the most defining impact on the nature of the data analytic architectures. Time bounds and constraints may require costly and specialized approaches to how is data stores and organized. We outline here three major temporally oriented types of analysis that fundamentally shape the characteristics of the data analysis and knowledge discovery architectures.

5.1.1 Platforms for Streaming Data Analysis

Data stream mining is a process of extraction of information and knowledge discovery from continuous, rapidly generated streams of data such as computer network traffic, system and usage logs, phone calls and conversations, ATM and credit card transactions, web searches, and sensor data. Data stream mining imposes unique constraints on the architectures usually manifested by a need for fast and often distributed analysis where analytic functions are pushed closer to the sources of data.

Stonebraker et al. [30] outline following three general architectural types for streaming data processing:

- Database Management Systems (DBMSs) - DBMS serve as collection and analytic points
- Rule engines - rule oriented software that executes set of rules on collected streams of data
- Stream Processing Engines (SPEs) - specialized stream processing software

Recent implementations of the stream data processing model at the large scale are Apache S4[28] implemented at Yahoo!, Apache Kafka[20] implemented at LinkedIn, Apache Flume[1], Twitter Storm[?] and Spark Streaming[?].

5.1.2 Platforms for Interactive Analysis

Interactive data analysis is a form of data analysis and knowledge discovery conducted interactively by a human user while exploring the data sets in a real time. This type of analysis is often facilitated via visualization tools, ad hoc queries, "what if" analysis and supported by data management systems capable of searching through and aggregating very large datasets.

Traditionally, architectures for interactive data analysis consisted of business intelligence tools attached to a relational datamart. Recently, this model has evolved to combine business intelligence tools with MPP databases. With emergence of "big data" technologies and visualization tools the architectural landscape has diversified. Several recent implementations have demonstrated use of alternative models for interactive analysis[25] including NoSQL implementations[27].

5.1.3 Platforms for Batch Oriented Analysis

Batch oriented analysis systems are the oldest, most established platforms for data analysis. Mainframe systems that still permeate markets[2] continue to perform analysis largely in a batch mode. Even the most recent "big data" platforms such as Hadoop operate in a batch mode. This

mode of operation is usually suitable for large data mining and machine learning tasks, and platforms such as Hadoop with Mahout offer low cost, high reliability alternative to costly mainframe architecture[17].

5.2 Storage and Data Representation Characterized Architectures

In this section we review some of the most recent trends in architectures influenced by physical organization of the storage and logical representation of data.

5.2.1 In-Memory Databases - Local and Distributed

In-memory databases rely primarily on main memory for data storage. In-memory databases run significantly faster than disk-oriented databases since these types of databases can remove transactional data structures and algorithms needed in hard disk based databases[18]. Currently, in-memory databases are used either as content and analytic data caching systems, or for data analytic operations, such as streaming data analysis, where low latency and response times are critical[24]. In-memory databases are often implemented as relational systems stripped of latching and logging mechanisms[31], or as a key-value, hash table oriented stores[34].

5.2.2 Disk Based, Distributed Databases

As we already discussed, all three major, large scale, data analytic architectures - Hadoop, Massively Parallel Processing Databases and High Performance Computing systems employ some form of massively parallel processing model for data analysis and computation. Lower cost of hardware, commoditization of high speed networks (10 GigE, Infiniband) and increases in both core counts and CPU clock speeds make previously prohibitively costly architectures feasible.[21] Within distributed architectures there are further classifications based on how is the shared storage organized and what are the associated consistency-availability-partitioning tradeoffs. [14]

5.2.3 Linked Data Oriented

Linked Data Oriented Architecture (LOA) is a logical, distributed data representation model that represents the data as collection of links, navigable via Uniform Resource Identifiers (URIs). LOA and related platforms are based on the principles of publishing on the web, originally outlined by Tim Berners-Lee[11] in his notes for Linked Data design:

- Use URIs as names for things
- Use HTTP URIs so that people can look up those names
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
- Include links to other URIs, so that they can discover more things

This type of architecture facilitates the process of knowledge discovery by enabling the machine or human user to navigate the graph of links and discover new relationships and

facts embedded in the network of links. Unlike distributed or in-memory data store models, Linked Oriented Architecture does not yet have formally established performance heuristics and architectural best practices for how should data be stored and organized. The assumption is that linked data based architectures offer greater flexibility of knowledge representation and ease of navigation of the knowledge graph than alternatives. Recent implementations of LOA-like architecture is Google's Knowledge Graph[5], DBPedia and Freebase.

5.3 Conclusion

With this paper we attempted to offer a broad survey of data analytic architectures with somewhat narrower focus on the general emerging trends and architectural taxonomies relevant for the effective implementation of data analytic operations. Field of data analysis and related data analytic architectures is continuously evolving, so we encourage further explorations and frequent surveys of the leading publications and conferences organized by ACM's special interest groups on data management systems (SIGMOD) and knowledge discovery from data (SIGKDD). In the domain of data analysis, industry and creative individuals are often advancing at the faster pace than research and academic communities, so we also recommend following trade conference such as Hadoop World, O'Reilly Strata and NoSQL Now!. High Availability[6] and myNoSQL[8] blogs are also excellent sources of information.

6. REFERENCES

- [1] Apache flume. <http://incubator.apache.org/flume>.
- [2] BMC software's sixth annual worldwide mainframe survey. <http://www.bmc.com/news/press-releases/2011/bmc-survey-shows-mainframe-use-is-still-strong.html> - accessed on July 15, 2012.
- [3] Greenplum database community edition.
- [4] HP Vertica database community edition.
- [5] Google knowledge graph, 2012. <http://goo.gl/sLsqp>, Retrieved on July 14, 2012.
- [6] High scalability, 2012. <http://highscalability.com/>.
- [7] Information visualization resources, 2012. <http://www.infovis.org>, accessed 14-July-2012.
- [8] mynosql, 2012. <http://nosql.mypopescu.com>.
- [9] D. Abadi. Tradeoffs between parallel database systems, hadoop, and hadoopdb as platforms for petabyte-scale analysis. In *Scientific and Statistical Database Management*, pages 1–3. Springer, 2010.
- [10] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin. Hadoopdb: An architectural hybrid of mapreduce and dbms technologies for analytical workloads. *Proceedings of the VLDB Endowment*, 2(1):922–933, 2009.
- [11] T. Berners-Lee. Design issues, linked data, 2006. <http://www.w3.org/DesignIssues/LinkedData.html>.
- [12] M. Beyer, D. Feinberg, M. Adrian, and R. Edjlali. Magic quadrant for data warehouse database management systems. *Gartner Research Note*, 2012.
- [13] D. Borthakur. Hdfs architecture guide, 2008.
- [14] E. Brewer. Towards robust distributed systems. In *Proceedings of the Annual ACM Symposium on*

- Principles of Distributed Computing*, volume 19, pages 7–10, 2000.
- [15] J. Dean. Challenges in building large-scale information retrieval systems: invited talk. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 1–1. ACM, 2009.
- [16] J. Dean and S. Ghemawat. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.
- [17] R. Esteves, R. Pais, and C. Rong. K-means clustering in the cloud—a mahout test. In *Advanced Information Networking and Applications (WAINA), 2011 IEEE Workshops of International Conference on*, pages 514–519. IEEE, 2011.
- [18] R. Fang, H. Hsiao, B. He, C. Mohan, and Y. Wang. High performance database logging using storage class memory. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pages 1221–1231. IEEE, 2011.
- [19] S. Ghemawat, H. Gobioff, and S. Leung. The google file system. In *ACM SIGOPS Operating Systems Review*, volume 37, pages 29–43, 2003.
- [20] K. Goodhope, J. Koshy, J. Kreps, N. Narkhede, R. Park, J. Rao, and V. Ye. Building linkedin’s real-time activity data pipeline. *Data Engineering*, page 33, 2012.
- [21] J. Hellerstein. The commoditization of massive data analysis, 2008.
- [22] KDnuggets.com. Survey - analytics, data mining, big data software used (may 2012). <http://www.kdnuggets.com/polls/2012/analytics-data-mining-big-data-software.html>, Accessed July 14, 2012.
- [23] R. Kimball and M. Ross. *The data warehouse toolkit: the complete guide to dimensional modeling*. Wiley, 2011.
- [24] P. Loos, J. Lechtenbörger, G. Vossen, A. Zeier, J. Krüger, J. Müller, W. Lehner, D. Kossmann, B. Fabian, O. Günther, et al. In-memory databases in business information systems. *Business & Information Systems Engineering*, pages 1–7, 2011.
- [25] S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton, and T. Vassilakis. Dremel: Interactive analysis of web-scale datasets. In *Proc. of the 36th Int’l Conf on Very Large Data Bases*, pages 330–339, 2010.
- [26] K. Monash. ebay’s two enormous data warehouses. <http://www.dbms2.com/2009/04/30/ebays-two-enormous-data-warehouses>, Accessed July 14, 2012.
- [27] P. Näsholm. Extracting data from nosql databases—a step towards interactive visual analysis of nosql data. 2012.
- [28] L. Neumeyer, B. Robbins, A. Nair, and A. Kesari. S4: Distributed stream computing platform. In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 170–177. IEEE, 2010.
- [29] M. Seeger and S. Ultra-Large-Sites. Key-value stores: a practical overview. *Computer Science and Media, Stuttgart*, 2009.
- [30] M. Stonebraker, U. Çetintemel, and S. Zdonik. The 8 requirements of real-time stream processing. *ACM SIGMOD Record*, 34(4):42–47, 2005.
- [31] M. Stonebraker, S. Madden, D. Abadi, S. Harizopoulos, N. Hachem, and P. Helland. The end of an architectural era:(it’s time for a complete rewrite). In *Proceedings of the 33rd international conference on Very large data bases*, pages 1150–1160. VLDB Endowment, 2007.
- [32] K. Whang. Nosql vs. parallel dbms for large-scale data management. 2011.
- [33] T. White. *Hadoop: The definitive guide. 3rd Ed.* Yahoo Press, 2012.
- [34] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. Technical report, Technical Report UCB/EECS-2011-82, EECS Department, University of California, Berkeley, 2011.