Big Data storage and Management: Challenges and Opportunities

J. Pokorný
Faculty of Mathematics and Physics,
Charles University, Prague
Czech Republic

Big Data Movement

- Something from Big Data Statistics
  - Facebook (2015) – generates about 10 TBytes every day
  - all Google data (2016): approximately 10EBytes
  - Twitter generates more than 7 TBytes every day
  - M. Lynch (1998): 80-90% of (business) data is unstructured
  - R. Birge (1996): memory capacity of the brain is ≈ 3 TB
  - The National Weather Service (2014): over 30 petabytes of new data per year (now over 3.5 billion observations collected per day)
- the digital universe is doubling in size every two years, and by 2020 – the data we create and copy annually – will reach 44 ZBytes or 44 trillion Gbytes

Big Data Movement

- Problem: our inability to utilize vast amounts of information effectively. It concerns:
  - data storage and processing at low-level (different formats)
  - analytical tools on higher levels (difficulties with data mining algorithms).
- Solution: new software and computer architectures for storage and processing Big Data including
  - new database technologies
  - new algorithms and methods for Big Data analysis, so called Big Analytics

Big Data Movement

On the other hand:

- J. L. Leidner[1] (R&D at Thompson Reuters, 2013): …
  - buzzwords like “Big Data” do not by themselves solve any problem – they are not magic bullets.
  - Advice: to solve any problem, look at the input data, specify the desired output data, and think hard about whether and how you can compute the desired result – nothing but “good old” computer science.

[1] interview with R. V. Zicari

Goal of the talk

- to present
  - some details of current database technologies typical for these (Big Data) architectures,
  - their pros and cons in different application environments,
  - their usability for Big Analytics, and
  - emerging trends in this area.

Content

- Big Data characteristics
- Big Data storage and processing
- NoSQL databases
- Apache Hadoop
- Big Data 2.0 processing systems
- Big Analytics
- Limitations of the Big Data
- Conclusions

Big Data „V“ characteristics

- Volume data at scale - size from TB to PB
- Velocity how quickly data is being produced and how quickly the data must be processed to meet demand analysis (e.g., streaming data)
  - Ex.: Twitter users are estimated to generate nearly 100,000 tweets every 60 sec.
- Variety data in many formats/media. There is a need to integrate this data together.

Big Data „V“ characteristics

- Veracity uncertainty/quality – managing the reliability and predictability of inherently imprecise data.
- Value worthwhile and valuable data for business (creating social and economic added value – see so called information economy).
- Visualization visual representations and insights for decision making.
- Variability the different meanings/contexts associated with a given piece of data (Forrester)

Big Data „V“ characteristics

- Volatility how long is data valid and how long should it be stored (at what point is data no longer relevant to the current analysis.
- Venue distributed, heterogeneous data from multiple platforms, from different owners’ systems, with different access and formatting requirements, private vs. public cloud.
- Vocabulary schema, data models, semantics, ontologies, taxonomies, and other content- and context-based metadata that describe the data’s structure, syntax, content, and provenance.

Big Data „V“ characteristics

- Vagueness Concerns a confusion over the meaning of Big Data. Is it Hadoop? Is it something that we’ve always had? What’s new about it? What are the tools? Which tools should I use? etc.
- Quality Quality characteristic measures how the data is reliable to be used for making decision.

Sometimes, a validity is considered. Similar to veracity, validity refers to how accurate and correct the data is for its intended use.

Big Data „V“ characteristics

Gardner´s definition (2001):

Big data is high-volume, -velocity and -variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making.

Remark: the first 3Vs are only 1/3 of the definition!

Big Data storage and processing

- general observation:
  - data and its analysis are becoming more and more complex
  - now: problem with data volume - it is a speed (velocity), not size!
  - necessity: to scale up and scale out both infrastructures and standard data processing techniques
- types of processing:
  - parallel processing of data in a distributed storage
  - real-time processing of data-in-motion
  - interactive processing and decision support processing of data-at-rest
  - batch oriented analysis (mining, machine learning, e-science)

Big Data storage and processing

- User options:
  - traditional parallel DBMS („shared-nothing“),
  - traditional distributed DBMS (DDBMS)
  - distributed file systems (GFS, HDFS)
  - programming models like MapReduce, Pregel
  - key-value data stores (so called NoSQL databases),
  - new architectures (New SQL databases).
- Applications are both transactional and analytical
  - they require usually different architectures

not in an operating systems sense

Towards scalable databases

- Features of traditional DBMS:
  - storage model
  - process manager
  - query processor
  - transactional storage manager
  - and shared utilities.

Towards scalable databases

- These technologies were transferred and extended into a parallel or distributed environment (DDBMS)
  - parallel or distributed query processing, distributed transactions (2PC protocol, …).
- Are they applicable in Big Data environment?
- Traditional DDBMS are not appropriate for Big Data storage and processing. They are many reasons for it, e.g.:
  - database administration may be complex (e.g. design, recovery),
  - distributed schema management,
  - distributed query management,
  - synchronous distributed concurrency control (2PC protocol) decreases update performance.

Scalability of DBMSs in context of Big Data

- Scalability. A system is scalable if increasing its resources (CPU, RAM, and disk) results in increased performance proportionally to the added resources.
- traditional scaling up (adding new expensive big servers)
  - requires higher level of skills
  - is not reliable in some cases

Scalability of DBMSs in context of Big Data

- Current architectural principle: scaling out (or horizontal scaling) based on data partitioning, i.e. dividing the database across many (inexpensive) machines
  - technique: data sharding, i.e. horizontal partitioning of data (e.g., hash or range partitioning)
  - compare: manual or user-oriented data distribution (DDBSs) vs. automatic data sharding (clouds, web DB, NoSQL DB)
- Data partitioning. Methods (1) vertical and (2) horizontal

  Ad (2)
  - Consistent hashing (Idea: the same hash function for both the object hashing and the node hashing)
  - Range partitioning (it is order-preserving)

Scalability of DBMSs in context of Big Data

- Consequences of scaling out:
  - scales well for both `reads` and `writes`
  - manage parallel access in the application
    - scaling out is not transparent, application needs to be partition-aware
  - influence on ACID guarantees
- „Big Data driven“ development of DBMSs
  - traditional solution: single server with very large memory and multi-core multiprocessor, e.g. HPC cluster, SSD storage, …
  - more feasible (network) solution: scaling-out with database sharding and replication

Big Data and Cloud Computing

- Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.[2]
- Cloud computing – its architecture and a way of data processing mean other way of data integration and dealing with Big Data.
- Cloud computing requires cloud databases.
- Ganz & Reinsel (2011): cloud computing accounts for less that 2% of IT spending (at 2011), by 2015, appr. 20% of information will be "touched" by a cloud computing service.

[2]Mell, P.,Grance, T.: The NIST Definition of Cloud Computing. NIST, 2011.

Scalable databases

- NoSQL databases,
- Apache Hadoop,
- Big Data Management Systems,
- NewSQL DBMSs,
- NoSQL databases with ACID transactions, and
- SQL-on-Hadoop systems.

NoSQL Databases

- The name stands for **N**ot **O**nly **SQL**
- NoSQL architectures differ from RDBMS in many key design aspects:
  - simplified data model,
  - database design is rather query driven,
  - integrity constraints are not supported,
  - there is no standard query language,
  - easy API (if SQL, then only its very restricted variant)
    - reduced access: CRUD operations – `create`, `read`, `update`, `delete`
    - no `join` operations (except within partitions),
    - no referential integrity constraints across partitions.

NoSQL databases

- Common features:
  - non-relational
  - usually do not require a fixed table schema
    $\Rightarrow$ more flexible data model
  - horizontally scalable
    $\Rightarrow$ scalability and performance advantages
  - replication support
  - relaxing ACID properties known from traditional DBMSs: Consistency, Avalability, Partition tolerance (see CAP theorem)
    - mostly AP systems

NoSQL databases

- Some other properties:
  - massive `write` performance (see, Facebook - 135 billion messages a month, Twitter - 7 TB/data per day)
  - fast key-value look-ups,
  - fast prototyping and development,
  - out of the box scalability,
  - easy maintenance,
  - mostly open source

Categories of NoSQL databases

- key-value stores
- column-oriented
- document-based
- graph databases (e.g. neo4j, InfoGrid - primarily focus on social network analysis)
- XML databases (e.g. myXMLDB, Tamino, Sedna)
- RDF databases (support the W3C RDF standard internally)

Categories of NoSQL databases

- key-value stores
- column-oriented
- document-based
- graph databases (e.g. neo4j, InfoGrid - primarily focus on social network analysis)
- XML databases (e.g. myXMLDB, Tamino, Sedna)
- RDF databases (support the W3C RDF standard internally)

Categories of NoSQL databases

- key-value stores

  Examples: SimpleDB, Redis

- column-oriented

  Examples: Cassandra, HBase

- document-based

  Examples: MongoDB, OrientDB , CouchDB

the most popular

Example: Key-Value Stores

key	(uniterpreted) value		
AR2673	Name = Jack	Grandchildren = Claire, Barbara, Magda	Nickname: Boy
D208HA	Name = Paul	Grandchildren = John, Ann	

Strenghts:

- simple data model
- simple scaling out horizontally (scalable, available)

Weaknesses:

- simplistic data model
- poor for complex data
- as the volume of data increases, maintaining unique values as keys may become more difficult

Typical features of NoSQL DB

- Parallelism: large data requires architectures to handle massive parallelism, scalability, reliability (no single point of failure),
- High performance: NoSQL solution is built for computing intensive applications, so high latency for any `read`/`write` operation is not accepted.
- Special indexing: efficient use of distributed indexes and main memory (RAM)

In terms of optimization, NoSQL DB can be divided into two categories:

- DB optimized for `read` operations (e.g., MongoDB, Redis, and OrientDB),
- DB optimized for `updates` (e.g., Cassandra and HBase).

Usability of NoSQL DBMSs (1)

- http://www.nosql-database.org/ lists currently > 225 NoSQL databases (including OO, Graph, XML, …)
- parts of data-intensive cloud apps (mainly Web apps).
  - Web entertainment applications,
  - indexing a large number of documents,
  - serving pages on high-traffic websites,
  - delivering streaming media,
  - data as typically occurs in social networking applications,
  - Examples:
    - Digg's 3 TB for green badges (markers that indicate stories upvoted by others in a social network)
    - Facebook's 50 TB for inbox search
    - Google uses BigTable in over 60 applications (e.g. Earth, Orkut)

Usability of NoSQL DBMSs (2)

- Applications do not requiring transactional semantics
  - address books, blogs, or content management systems
  - analyzing high-volume, real time data (e.g., Web-site click streams)
  - mobile computing makes transactions at large scale technically infeasible

$\Rightarrow$

- Enforcing schemas and row-level locking as in RDBMS — unnecessarily over-complicate these applications.
- Moreover: absence of ACID allows significant acceleration and decentralization of NoSQL databases

Usability of NoSQL DBMSs (3)

- Unusual and often inappropriate phenomena in NoSQL approaches:
  - have little or no use for data modeling
  - developers generally do not create a logical model
  - query driven database design
  - unconstrained data
  - different behavior in different applications
  - no query language standard
  - complicated migration from one such system to another.

Usability of NoSQL DBMSs (4)

- Examples of problems with NoSQL
  - Hadoop stack: poor performance except where the application is „trivially parallel“
    - reasons: no indexing, very inefficient joins in Hadoop layer, „sending the data to the query“ and not „sending the query to the data“
  - Couchbase: replications of the whole document if only its small part is changed.
- Inadvisability of NoSQL for
  - most of the DW and BI querying
    - few facilities for ad-hoc query and analysis. Even a simple query requires significant programming expertise, and commonly used BI tools do not provide connectivity to NoSQL.
    - E.g., HBase – fast analytical queries, but on column level only
  - applications requiring enterprise-level functionality (ACID, security, and other features of RDBMS technology).
  - NoSQL should not be the only option in the cloud.

DB-Engines Ranking*

- NoSQL DBMSs are significant in the Database World!

328 systems in ranking, May 2017

Rank	DBMS	Database Model	Score
1.	Oracle	Relational	1354.31
2.	MySQL	Relational	1340.03
3.	Microsoft SQL Server	Relational	1213.80
4	PostgreSQL	Relational	365.91
5.	MongoDB	Document store	331.58
6.	DB2	Relational	188.84
7.	Microsoft Access	Relational	129.87
8.	Cassandra	Wide column store	123.11
10.	Redis	Key-value	117.45
9.	SQLite	Relational	116.07

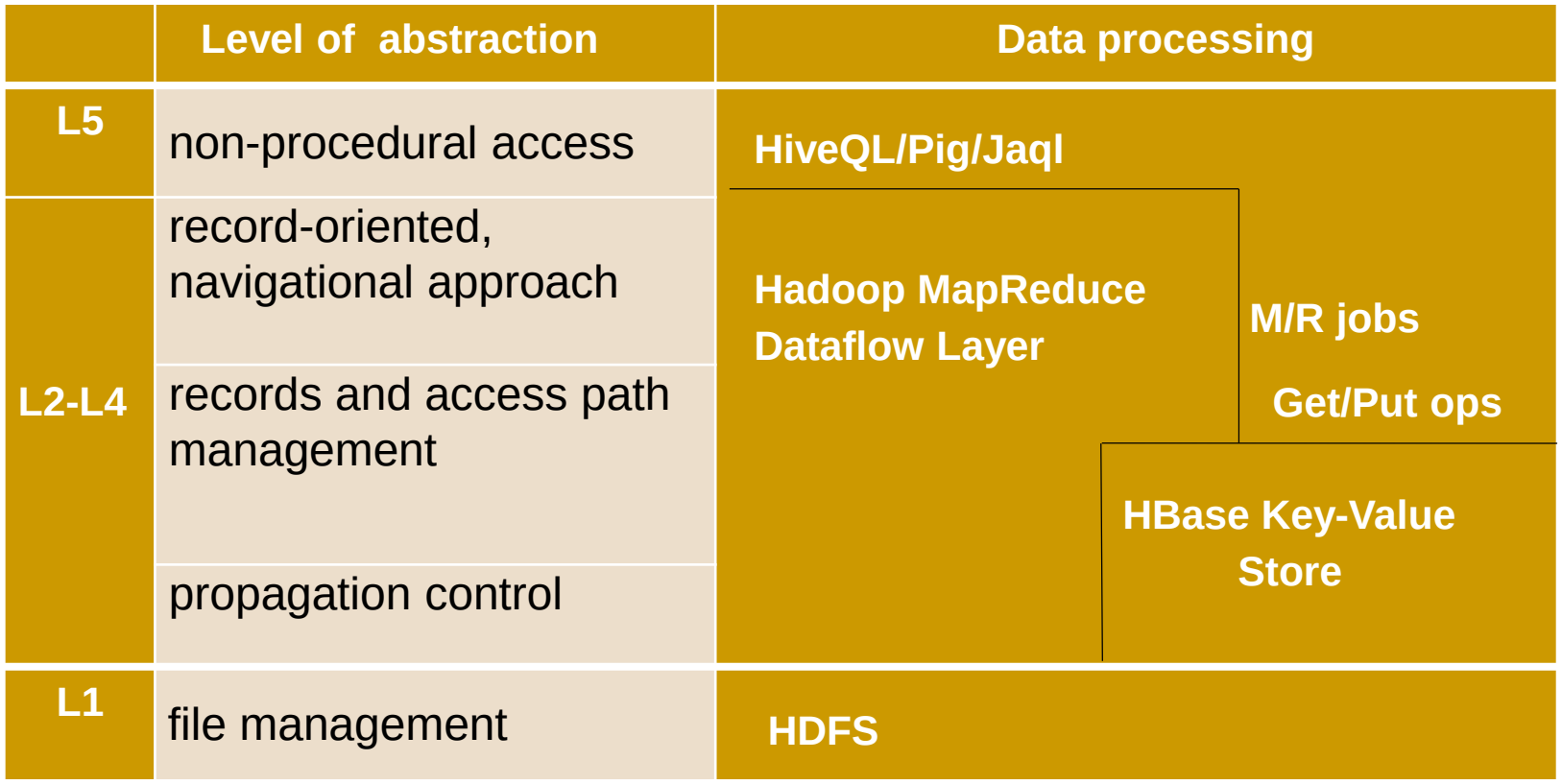
*http://db-engines.com/en/ranking

Apache Hadoop

- Hadoop: a batch processing Big Data infrastructure platform.
  - Data is stored in files managed by the Hadoop Distributed File System (HDFS).
  - Parallelized distributed computing on server clusters is done by a highly popular infrastructure and programming model MapReduce (MR).
  - Example of software: Apache Hive - an open-source data warehouse (DW) system for querying and analyzing large datasets stored in Hadoop files.
  - Extensions to SQL: SQL-like language variant HiveQL implemented in Apache Hive.
  - Many NoSQL databases use Hadoop for their data.

Example: Hadoop software stack

in version Apache Software Foundation – data access through more layers



Big Data 2.0 Processing Systems

- Older categorization: NewSQL DBMS, NoSQL with ACID transactions, …
- From 2016 (Bajaber, Sakr):
  - General Purpose Big Data Processing Systems
    - Often Big Data Managements Systems
  - Big SQL Processing Systems
    - Often NewSQL DBMSs
  - Big Graph Processing Systems
  - Big Stream Processing Systems

BDMS

- ASTERIX (Vinayak et al, 2012) - uses fuzzy matching for analytical purposes
- Oracle Big Data Appliance (2011)
  - Oracle Big Data SQL (combines data from Oracle DB, Hadoop and NoSQL in a single SQL query; enables to query and analyze data in Hadoop and NoSQL)
  - Oracle Big Data Connectors (for simplifying data integration and analytics of data from Hadoop)
  - Includes: Oracle Exadata Database Machine and Oracle Exalytics Business Intelligence Machine
- Lily (NGDATA, 2014)
  - integrates Apache Hadoop, HBase, Solr and machine learning

NewSQL DBMS

- Next generation of highly scalable and elastic RDBMS: NewSQL databases (from April 2011):
  - designed to scale out horizontally on shared nothing machines,
  - still provide ACID guarantees,
  - applications interact with the DB primarily using SQL (with joins),
  - employ a lock-free concurrency control,
  - provide higher performance than available from the traditional systems.

NewSQL DBMS

- General purpose distributed DBMSs: ClustrixDB, NuoDB (appropriate for clouds), VoltDB
- Google: Spanner and F1
  - Spanner uses semi-relations, i.e. each row has a name (i.e. always there is a primary key), versions, hierarchies of tables
  - F1 is built on Spanner
- Hadoop-relational hybrids (e.g. HadoopDB - a parallel DB with Hadoop connectors, Vertica)

NewSQL DBMS

- Transparent sharding: MySQL Cluster, (ClustrixDB), ScaleArc, …
- In-memory DBMS: MemSQL (MySQL-like), VoltDB
- Postgres with NoSQL features:
  - native datatypes JSON and HSTORE in SQL
    - HSTORE value contains a set of key-value couples

NoSQL with ACID transactions

- A new generation of NoSQL databases.
  - maintain distributed design, fault tolerance, easy scaling, and a simple, flexible base data model,
  - they are CP systems with global transactions,
  - extend the base data models of NoSQL
- FoundationDB is a key-value store with scalability and fault tolerance (and an SQL layer).
- MarkLogic is a NoSQL document database with JSON storage, HDFS, optimistic locking
- OrientDB is a Distributed Graph Database

Current state - practice

- Problems with cloud computing:
  - SaaS applications require enterprise-level functionality, including ACID transactions, security, and other features associated with commercial RDBMS technology, i.e. NoSQL should not be the only option in the cloud.
- A middle-road: adapting AP practices to a commercial RDBMS
  - eBay with Oracle (includes restrictions like: tables were denormalized to a large extent, transactions were forbidden with few exceptions, joins, Group Bys, and Sorts were done in the application layer

Big Analytics

- New alternatives for Big Analytics with NewSQL:
  - HadoopDB (combines parallel database with Map Reduce)
  - MemSQL, VoltDB (automatic cross-partition joins), but performance is still a question
  - ClustrixDB (for TP and real-time analytics)
- Near future:
  - Big data: The future is in analytics!
  - shipping to „No Hadoop“ DBMSs (MapReduce layer is eliminated)
  - Towards Analytics 3.0: new wave of Big Analytics,
    - Analytics 1.0 is BI
    - Analytics 2.0, which is used by online companies only (Google, Yahoo, Facebook, etc.)

Limitations of the Big Data

- Collecting and analyzing data from any real world process must follow the same principles in statistical study design and data analysis.
- Bigger data is not always better data.
  - Quantity does not necessarily means quality, see, e.g., data from social networks.
  - Big sample size does not remove bias (<-sampling)
- Big Data is prone to data errors.
  - Sometimes errors or bias are undetected owing to the size of the sample and thus produce inaccurate results.
- Big Analytics is often subjective.
  - There can be multiple ways to look at the same information and to interpret it differently by different users.

Limitations of the Big Data

- Not all the data is useful.
  - i.e., collecting data which is never used or which does not answer a particular question is relatively useless.
  - only a small subset is interesting to us - find a needle in a hay stack (dimension reduction, Google's MapReduce, real time data analysis).
- Accessing Big Data raises ethical issues.
  - Both in industry and in academics the issues of privacy and accountability with respect to Big Data have now raised important concerns.
- Big Data creates a new type of digital divide.
  - Having access and knowledge of Big Data technologies gives companies and people a competitive edge in today's data driven world.

Conclusions

10 hottest Big Data technologies based on Forrester's analysis from 2016:

- continuing development of NoSQL databases,
- distributed datastores,
- in-memory data fabric (dynamic random access memory, flash, or SSD),
- data preparation (sourcing, shaping, cleansing, and sharing diverse and messy data sets),
- data quality (product that conduct data cleansing, …)
- data virtualization (delivering information from various data sources in real-time or near-real time)
- data integration should contribute to delivering in-formation from various data sources and to data orchestration across various exiting solutions (Hadoop, NoSQL, Spark, etc.).

Conclusions

The rest:

- predictive analysis (to discover, evaluate, optimize, and deploy predictive models by analyzing Big Data sources to improve business performance or mitigate risk),
- search and knowledge discovery (o support self-service extraction of information and new insights from large repositories), and
- stream analytics (filter, aggregate, enrich, and analyze a high throughput of data from multiple disparate live data sources and in any data format)

should support Big Analytics applications.

But! The biggest challenge does not seem the technology itself. More important problem is, how to have enough skills to make effective use of these technologies at disposal and make sense out of the data collected.