In-Memory OLAP Reading List

Yifu Huang

Large Scale OLAP

Pig [http://pig.apache.org/] is developed by Yahoo!.

What:

A new language called Pig Latin is designed to fit in a sweet spot between the declarative style of SQL, and the low-level, procedural style of map-reduce. A accompanying system called Pig compiles Pig Latin into physical plans that are executed over Hadoop. A novel debugging environment called Pig Pen leads to even higher productivity gains.

Why:

Parallel databases are usually prohibitively expensive at large scale. The declarative SQL style is unnatural. The map-reduce paradigm is too low-level and rigid.

How:

The nested data model is used, and Pig Latin is designed as a dataflow language. Pig builds a logical plan for a Pig Latin Program, and then compiles a logical plan into map-reduce jobs executed using Hadoop. Pig Pen creates a side data set automatically to avoid the inefficiency of constructing a Pig Latin program.

References:

[1] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, Andrew Tomkins: Pig latin: a not-so-foreign language for data processing. SIGMOD 2008:1099-1110

Notes:

There is a family of recent, large-scale, distributed systems that provide database-like capabilites, but are geared toward transactional workloads and/or point lookups. Amazon's Dynamo, Google's BigTable, and Yahoo!'s PNUTS are examples. Sawzall is a scripting language used at Google on top of map-reduce.

SCOPE is developed by Microsoft.

What:

A new declarative and extensible scripting language, SCOPE(Structured Computations Optimized for Parallel Execution), is targeted for massive data analysis.

Why:

It is imperative to develop a programming model that hides the complexity of the underlying system but provides flexibility by allowing users to extend functionality to meet a variety of requirements.

How:

SCOPE borrows several features from SQL, and new operators(extractors, processors, reducers, combiners) are also implemented. So SCOPE supports nesting of expressions but also allows a computation to be specified as a series of steps. Scripts are compiled into efficient, parallel execution plans and executed on large clusters.

References:

[1] Ronnie Chaiken, Bob Jenkins, Per-Åke Larson, Bill Ramsey, Darren Shakib, Simon Weaver, Jingren

Zhou: SCOPE: easy and efficient parallel processing of massive data sets. PVLDB 1(2):1265-1276 (2008)

Notes:

Several companies have developed distributed data storage and processing systems on large clusters of shared-nothing commodity servers, including Google's File System, Bigtable Map-Reduce, Hadoop, Yahoo!'s Pig system Ask.com's Neptune, and Microsoft's Dryad. Both SCOPE and DryadLINQ use Dryad as a runtime. Microsoft has developed a distributed computing platform, called Cosmos, for storing and analyzing massive data sets.

Hive [http://hive.apache.org/] is developed by Facebook.

What:

A data warehousing solution is built on top of Hadoop which supports HiveQL(a SQL-like declarative language) for OLAP.

Why:

Traditional warehousing solutions are prohibitively expensive. The map-reduce programming model is very low level.

How:

HiveQL is compiled into the execution plan consists of a directed-acyclic graph (DAG) of map-reduce jobs.

References:

[1] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, Raghotham Murthy: Hive - A Warehousing Solution Over a Map-Reduce Framework. PVLDB 2(2):1626-1629 (2009)

[2] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Anthony, Hao Liu, Raghotham Murthy: Hive - a petabyte scale data warehouse using Hadoop. ICDE 2010:996-1005

HadoopDB [http://db.cs.yale.edu/hadoopdb/hadoopdb.html] is developed by Yale.

What:

A hybrid system integrated with MapReduce-based systems (superior scalability, fault tolerance, and flexibility) and parallel databases (high performance and efficiency) for OLAP.

Why:

MapReduce-based systems scale well but lack many of the features for structured data analysis workloads, which cause slow performance. For OLAP, parallel databases can achieve high performance and efficiency, but do not scale well.

How:

Use MapReduce as the communication layer above multiple nodes running single-node DBMS instances. In details, use Hive as the translation layer, Hadoop as the communication layer and PostgreSQL as the database layer.

References:

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

[2] Azza Abouzied, Kamil Bajda-Pawlikowski, Jiewen Huang, Daniel J. Abadi, Avi Silberschatz:

HadoopDB in action: building real world applications. SIGMOD 2010:1111-1114
Notes:
HadoopDB is commercialized into product Hadapt [http://hadapt.com/]. Companies in recent years build specialized analytical data management software(e.g., Netezza, Vertica, DATAllegro, Greenplum, Aster Data, Infobright, Kickfire, Dataupia, ParAccel, and Exasol).

Dremel is developed by Google.
What:
Dremel is a scalable, interactive ad-hoc query system for analysis of read-only nested data.
Why:
Performing interactive data analysis at scale demands a high degree of parallelism. The data used in web and scientific computing is often non-relational.
How:
Dremel builds on ideas from web search and parallel DBMSs. By combining multi-level execution trees and columnar data layout, it is capable of running aggregation queries over trillion-row tables in seconds.
References:
[1] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, Theo Vassilakis: Dremel: Interactive Analysis of Web-Scale Datasets. PVLDB 3(1):330-339 (2010)
Notes:
In contrast to layers such as Pig and Hive, Dremel executes queries natively without translating them into MR jobs. The columnar storage format is supported by many data processing tools at Google, including MR, Sawzall, and FlumeJava. Drill [http://incubator.apache.org/drill/] is the open source implementation of Dremel. Impala [http://impala.io/] is developed by Cloudera. A number of hybrid systems that combine parallel DBMSs with MR, are offered by vendors
like Aster, Cloudera, Greenplum, and Vertica.

In-Memory OLAP

SAP NetWeaver Business Warehouse Accelerator is developed by SAP.
What:
An example of an in-memory OLAP engine with a focus on storage architecture.
Why:
Data storage for in-memory OLAP engines is often implemented in standard storage technologies like storage area network (SAN) or network attached storage (NAS) with high hardware costs.
How:
Given the access pattern, storage costs can be reduced by using a distributed persistence layer based on commodity architecture.
References:
[1] Olga Mordvinova, Oleksandr Shepil, Thomas Ludwig, Andrew Ross: A strategy for cost efficient distributed data storage for in-memory OLAP. IADIS AC 2009:109-117
Notes:

Tables of data are split vertically into separate columns, each listing values for one attribute, which is used by C-Store, MonetDB, and Vertica. The idea of inexpensive commodity storage has been explored in the industrial (Google FS, Hadoop FS, FARSITE, Vertica), scientific (Ceph, XtreemFS, OceanStore), and high-performance (Lustre) communities.

What:
Techniques are used for efficient execution of analytical SQL queries on large amounts of data in a parallel database cluster while making maximal use of the available hardware.
Why:
There has been little work about in-memory OLAP in the context of clusters.
How:
This includes precompiled query plans for efficient CPU utilization, full parallelization on single nodes and across the cluster, and efficient inter-node communication.
References:
[1] Martin Weidner, Jonathan Dees, Peter Sanders: Fast OLAP query execution in main memory on large data in a cluster. BigData Conference 2013:518-524
Notes:
There are other databases like HyPer or DBToaster, which have shown that performing just-in-time compilation of SQL queries into native code is possible.

Hybrid In-Memory OLTP&OLAP

What:
A common database approach for integrated OLTP and OLAP with high level scope.
Why:
While centralized warehouses also handle the integration of data from many sources, it is still desirable to have OLTP and OLAP capabilities in one system which could make both components more valuable to their users. Column storage was successfully used for many years in OLAP and really surged when main memory became abundant.
How:
Using in-memory column databases.
[1] Hasso Plattner: A common database approach for OLTP and OLAP using an in-memory column database. SIGMOD 2009:1-2

Hyper [http://hyper-db.de/] is developed by TUM.
What:
An efficient hybrid system, called HyPer, can handle both OLTP and OLAP simultaneously. ScyPer is a version of the HyPer that horizontally scales out on sharednothing commodity hardware.
Why:
While allowing for decent transaction rates, the separation of OLTP and OLAP has many disadvantages including data freshness issues due to the delay caused by only periodically initiating the Extract Transform Load-data staging and excessive resource consumption due to maintaining

two separate information systems.
How:
Using hardware-assisted replication mechanisms to maintain consistent snapshots of the transactional data.
References:
[1] Alfons Kemper, Thomas Neumann: HyPer: A hybrid OLTP&OLAP main memory database system based on virtual memory snapshots. ICDE 2011:195-206
[2] Alfons Kemper, Thomas Neumann, Jan Finis, Florian Funke, Viktor Leis, Henrik Mühe, Tobias Mühlbauer, Wolf Rödiger: Processing in the Hybrid OLTP & OLAP Main-Memory Database System HyPer. IEEE Data Eng. Bull. (DEBU) 36(2):41-47 (2013)
[3] Tobias Mühlbauer, Wolf Rödiger, Angelika Reiser, Alfons Kemper, Thomas Neumann: ScyPer: A Hybrid OLTP&OLAP Distributed Main Memory Database System for Scalable Real-Time Analytics. BTW 2013:499-502
Notes:
There are dedicated OLTP main memory systems such as VoltDB or TimesTen. There are dedicated OLAP main memory DBMS such as MonetDB or TREX.

In-Memory OLTP

H-Store [http://hstore.cs.brown.edu/] is developed by MIT, Brown, CMU, Yale and Intel.
What:
A high performance main memory database that operates on a distributed cluster of shared-nothing machines for OLTP.
Why:
OLTP is hindered by the I/O performance of legacy RDBMS platforms. Using a disk-oriented RDBMS is another key bottleneck in OLTP databases. The availability of multiple-cores, the abundance of main memory, the lack of user stalls, and the dominant use of stored procedures are factors that portend a clean-slate redesign of RDBMSs.
How:
Partitioning across multiple shared-nothing machines. Use a main memory database.
References:
[1] Robert Kallman, Hideaki Kimura, Jonathan Natkins, Andrew Pavlo, Alex Rasin, Stanley B. Zdonik, Evan P. C. Jones, Samuel Madden, Michael Stonebraker, Yang Zhang, John Hugg, Daniel J. Abadi: H-store: a high-performance, distributed main memory transaction processing system. PVLDB 1(2):1496-1499 (2008)
[2] Michael Stonebraker, Ariel Weisberg: The VoltDB Main Memory DBMS. IEEE Data Eng. Bull. (DEBU) 36(2):21-27 (2013)
Notes:
H-Store is commercialized into product VoltDB [http://voltdb.com/].

In-Memory Cloud Computing

Spark [http://spark.apache.org/] is developed by UCB.

What:

Resilient Distributed Datasets(RDDs) is a distributed memory abstraction that lets programmers perform in-memory computations on large clusters in a fault-tolerant manner. RDDs is implemented in a system called Spark.

Why:

RDDs are motivated by two types of applications that current computing frameworks handle inefficiently: iterative algorithms and interactive data mining tools.

How:

To achieve fault tolerance efficiently, RDDs provide a restricted form of shared memory, based on coarsegrained transformations rather than fine-grained updates to shared state.

References:

[1] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauly, Michael J. Franklin, Scott Shenker, Ion Stoica: Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. NSDI 2012:15-28

Notes:

Data flow models such as MapReduce, Dryad and Ciel support a rich set of operators for processing data but share it through stable storage systems. High-level programming interfaces for data flow systems, including DryadLINQ and FlumeJava, provide language-integrated APIs where the user manipulates 'parallel collections' through operators like map and join. Pregel supports iterative graph applications, while Twister and HaLoop are iterative MapReduce runtimes. Piccolo lets users run parallel functions that read and update cells in a distributed hash table. Distributed shared memory(DSM) systems and key-value stores like RAMCloud offer a similar model. Nectar can reuse intermediate results across DryadLINQ jobs by identifying common subexpressions with program analysis. Ciel and FlumeJava can likewise cache task results but do not provide in-memory caching or explicit control over which data is cached.

Distributed Databases

Spanner is developed by Google.

What:

Spanner is Google's scalable, multi-version, globallydistributed, and synchronously-replicated database.

Why:

There is no system to distribute data at global scale and support externally-consistent distributed transactions.

How:

It is a database that shards data across many sets of Paxos state machines in datacenters spread all over the world.

References:

[1] James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J. J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, Wilson C. 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, Dale Woodford: Spanner: Google's Globally-Distributed Database. OSDI 2012:261-264
Notes:
F1 is a rewrite of Google's advertising backend. Megastore has semi-relational data model and supports for synchronous replication, but has relatively poor write throughput. Colossus is the successor to the Google File System. Percolator was in part built to address the lack of cross-row transactions in Bigtable. Consistent replication across datacenters as a storage service has been provided by Megastore and DynamoDB. Scatter is a recent DHT-based key-value store that layers transactions on top of consistent replication. H-Store and Granola each supported their own classification of transaction types, some of which could avoid locking. VoltDB is a sharded in-memory database that supports master-slave replication over the wide area for disaster recovery, but not more general replication configurations. It is an example of what has been called NewSQL, which is a marketplace push to support scalable SQL. A number of commercial databases implement reads in the past, such as MarkLogic and Oracle's Total Recall.
