Introduction to Apache Drill

February 27, 2013

Tomer Shiran
Who Am I?

- Tomer Shiran
- tshiran@maprtech.com
- Founding Member and Committer, Apache Drill
- Director of Product Management, MapR Technologies
Agenda

• Apache Drill overview
• Key features
• Status and progress
• How to get involved
Big Data Workloads

- ETL
- Data mining
- Index and model generation
- Clustering, anomaly detection and classification
- Blob store
- Lightweight OLTP on large datasets
- Web crawling
- Stream processing
- Interactive analysis
Interactive Queries and Hadoop

Existing Solutions

- **Hive**: Compile SQL (HiveQL) to MapReduce
- **Impala**: Export MapReduce results to RDBMS and query RDBMS

Emerging Technologies

- **Apache Drill**
  - Real-time interactive analysis
- **HADAPT**
  - PostgreSQL-based Hadoop analytics
- **citusdata**
  - PostgreSQL-based Hadoop analytics
- **Stinger/Tez**
  - Hive performance improvements
- **Impala**
  - Real-time interactive analysis
- **Greenplum HAWQ**
  - PostgreSQL-based Hadoop analytics
- **Phoenix**
  - SQL layer for HBase
- **Cascading Lingual**
  - Compile ANSI SQL to MapReduce
Example Problem

• Jane works as an analyst at an e-commerce company
• How does she figure out good targeting segments for the next marketing campaign?
• She has some ideas and lots of data
Solving the Problem with Traditional Systems

• Use an RDBMS
  – ETL the data from MongoDB and Hadoop into the RDBMS
    • MongoDB data must be flattened, schematized, filtered and aggregated
    • Hadoop data must be filtered and aggregated
  – Query the data using any SQL-based tool

• Use MapReduce
  – ETL the data from Oracle and MongoDB into Hadoop
  – Work with the MapReduce team to generate the desired analyses

• Use Hive
  – ETL the data from Oracle and MongoDB into Hadoop
    • MongoDB data must be flattened and schematized
  – But HiveQL is limited, queries take too long and BI tool support is limited
### WWGD

Build Apache Drill to provide a true open source solution to interactive analysis of Big Data

<table>
<thead>
<tr>
<th></th>
<th>Distributed File System</th>
<th>NoSQL</th>
<th>Interactive analysis</th>
<th>Batch processing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Google</strong></td>
<td>GFS</td>
<td>BigTable</td>
<td>Dremel</td>
<td>MapReduce</td>
</tr>
<tr>
<td><strong>Apache</strong></td>
<td>HDFS</td>
<td>HBase</td>
<td>???</td>
<td>Hadoop MapReduce</td>
</tr>
</tbody>
</table>
Apache Drill Overview

- Interactive analysis of Big Data using standard SQL
- Fast
  - Low latency
  - Columnar execution
    • Inspired by Google Dremel/BigQuery
  - Complement native interfaces and MapReduce/Hive/Pig
- Open
  - Community driven open source project
  - Under Apache Software Foundation
- Modern
  - Standard ANSI SQL:2003 (select/into)
  - Nested/hierarchical data support
  - Schema is optional
  - Supports RDBMS, Hadoop and NoSQL

Interactive queries
Data analyst Reporting
100 ms-20 min

MapReduce
Hive
Pig

Data mining
Modeling
Large ETL
20 min-20 hr
How Does It Work?

- Drillbits run on each node, designed to maximize data locality
- Processing is done outside MapReduce paradigm (YARN is supported)
- Queries can be fed to any Drillbit
- Coordination, query planning, optimization, scheduling, and execution are distributed

SELECT * FROM oracle.transactions, mongo.users, hdfs.events LIMIT 1
Key Features

- Full SQL (ANSI SQL:2003)
- Nested data
- Schema is optional
- Flexible and extensible architecture
Full SQL (ANSI SQL:2003)

• Drill supports standard ANSI SQL:2003
  – Correlated subqueries, analytic functions, ...
  – SQL-like is not enough

• Use any SQL-based tool with Apache Drill
  – Tableau, Microstrategy, Excel, SAP Crystal Reports, Toad, SQuirreL, ...
  – Standard ODBC and JDBC drivers
Nested Data

- Nested data is becoming prevalent
  - JSON, BSON, XML, Protocol Buffers, Avro, ...
  - The data source may or may not be aware
    • MongoDB supports nested data natively
    • A single HBase value could be a JSON document (compound nested type)
  - Google Dremel’s innovation was efficient columnar storage and querying of nested data

- Flattening nested data is error-prone and often impossible
  - Think about repeated and optional fields at every level...

- Apache Drill supports nested data
  - Extensions to ANSI SQL:2003

```json
{
  "name": "Homer",
  "gender": "Male",
  "followers": 100
}
```

```avro
enum Gender {
  MALE, FEMALE
}

record User {
  string name;
  Gender gender;
  long followers;
}
```
Schema is Optional

- Many data sources do not have rigid schemas
  - Schemas change rapidly
  - Each record may have a different schema
    - Sparse and wide rows in HBase and Cassandra, MongoDB
- Apache Drill supports querying against unknown schemas
  - Query any HBase, Cassandra or MongoDB table
- User can define the schema or let the system discover it automatically
  - System of record may already have schema information
    - Why manage it in a separate system?
  - No need to manage schema evolution

<table>
<thead>
<tr>
<th>Row Key</th>
<th>CF contents</th>
<th>CF anchor</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;com.cnn.www&quot;</td>
<td>contents:html = &quot;&lt;html&gt;...&quot;</td>
<td>anchor:my.look.ca = &quot;CNN.com&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>anchor:cnnsi.com = &quot;CNN&quot;</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Flexible and Extensible Architecture

- Apache Drill is designed for extensibility
  - Well-documented APIs and interfaces
- Data sources and file formats
  - Implement a custom scanner to support a new data source or file format
- Query languages
  - SQL:2003 is the primary language
  - Implement a custom Parser to support a Domain Specific Language
  - UDFs and UDTFs
- Optimizers
  - Drill will have a cost-based optimizer
  - Clear surrounding APIs support easy optimizer exploration
- Operators
  - Custom operators can be implemented
    - Special operators for Mahout (k-means) being designed
  - Operator push-down to data source (RDBMS)
What About Other SQL-on-Hadoop Systems?

• Strengths
  – Code is more mature than Apache Drill
    • Already in beta (~2 quarters ahead)
  – Faster than Hive on some queries

• Weaknesses
  – Proprietary or semi-open source
  – Query results must fit in memory (no spooling)
  – Early row materialization (no columnar execution)
  – Some are SQL-like (not SQL)
  – No support for nested data
  – Rigid schema is required
  – Limited flexibility and extensibility
  – Only support Hadoop and HBase (and no other NoSQL or RDBMS)
Status: In Progress

- Heavy active development
  - 6-7 companies are contributing
- Available
  - Logical plan syntax and interpreter
  - Reference execution engine
- In progress
  - SQL interpreter
  - Storage engine implementations for Accumulo, Cassandra, HBase and various file formats
- Significant community momentum
  - Over 250 people on the Drill mailing list
  - Over 250 members of the Bay Area Drill User Group
  - Drill meetups across the US and Europe
  - OpenDremel team joined Apache Drill
- Anticipated schedule:
  - Prototype: Q1
  - Alpha: Q2
  - Beta: Q3
Why Apache Drill Will Be Successful

Resources
• Contributors have strong backgrounds from companies like Oracle, IBM Netezza, Informatica, Clustrix and Pentaho

Community
• Development done in the open
• Active contributors from multiple companies
• Rapidly growing

Architecture
• Full SQL
• New data support
• Extensible APIs
• Full Columnar Execution
• Beyond Hadoop
Interested in Apache Drill?

• Many options to contribute
  – Become a full time Drill engineer @ MapR
    • Email tshiran@maprtech.com
  – Join the Drill mailing list and start contributing
    • JIRAs, code, unit tests, documentation, ...
  – Shoot me an email and we can discuss
    • Email tshiran@maprtech.com
QUESTIONS?
Why Not Leverage MapReduce?

- **Scheduling Model**
  - Coarse resource model reduces hardware utilization
  - Acquisition of resources typically takes 100’s of millis to seconds

- **Barriers**
  - Map completion required before shuffle/reduce commencement
  - All maps must complete before reduce can start
  - In chained jobs, one job must finish entirely before the next one can start

- **Persistence and Recoverability**
  - Data is persisted to disk between each barrier
  - Serialization and deserialization are required between execution phase