

Apache MRQL - MapReduce Query Language

MARK MCCOMBE^{1,*}

¹ School of Informatics and Computing, Bloomington, IN 47408, U.S.A.

* Corresponding authors: mmccombe@iu.edu

project-000, March 26, 2017

Apache Map Reduce Query Language (MRQL) is a project currently in the Apache Incubator. MRQL runs on Apache Hadoop, Hama, Spark, and Flink. An overview of each dependent technology is provided along with an outline of its architecture. MRQL and MapReduce are introduced, along with a look at the MRQL language. Alternative technologies are discussed with Apache Hive and Pig highlighted. Finally, use cases for MRQL and resources for learning more about the project are presented.

© 2017 <https://creativecommons.org/licenses/>. The authors verify that the text is not plagiarized.

Keywords: MRQL, MapReduce, Apache, Hadoop, Hama, Spark, Flink, I524

<https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IO-3012/report.pdf>

INTRODUCTION

Apache MapReduce Query Language (MRQL) "is a query processing and optimization system for large-scale, distributed data analysis" [1]. MRQL provides a SQL like language for use on Apache Hadoop, Hama, Spark, and Flink. MapReduce Query Language allows users to perform complex data analysis using only SQL like queries, which are translated by MRQL into efficient Java code, removing the burden of writing MapReduce code directly. MRQL can evaluate queries in Map-Reduce (using Hadoop), Bulk Synchronous Parallel (using Hama), Spark, and Flink modes [1].

MRQL was created in 2011 by Leontidas Fegaras [2] and is currently in the Apache Incubator. All projects accepted by the Apache Software Foundation (ASF) undergo an incubation period until a review indicates that the project meets the standards of other ASF projects [3]. MRQL is pronounced "miracle" [1].

ARCHITECTURE

MRQL can run on top of Apache Hadoop, Apache Hama, Apache Spark, and Apache Flink clusters. The architecture of each technology is discussed in the following sections along with the architecture of MRQL itself.

Apache Hadoop

Apache Hadoop is an open source framework written in Java that utilizes distributed storage and the MapReduce programming model for processing of big data. Hadoop utilizes commodity hardware to build fault tolerant clusters [4].

As show in Figure 1, Hadoop consists of several building blocks: the Cluster, Storage, Hadoop Distributed File System (HDFS) Federation, Yarn Infrastructure, and the MapReduce Framework. The Cluster is comprised of multiple machines,

otherwise referred to as nodes. Storage can be in the HDFS or an alternative storage medium such as Amazon Web Service's Simple Storage Service (S3). HDFS federation is the framework responsible for this storage layer. YARN Infrastructure provides computational resources such as CPU and memory. The MapReduce layer is responsible for implementing MapReduce [5]. Additionally, Hadoop includes the Hadoop Common Package (not shown in Figure 1), which includes operating and file system abstractions and JAR files needed to start Hadoop [4].

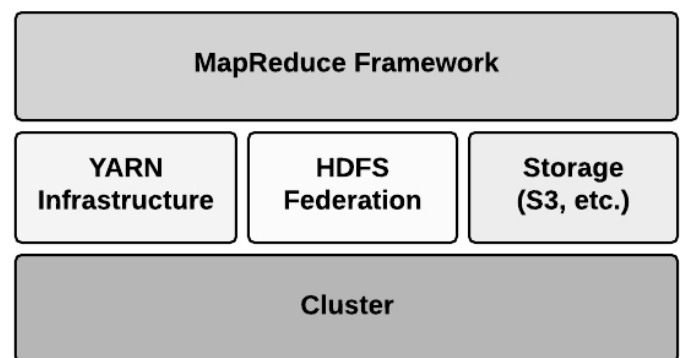


Fig. 1. Hadoop Components [5]

Figure 2 depicts a simple multi-node Hadoop cluster. The cluster contains master and slave nodes. The master node contains a DataNode, a NameNode, a Job Tracker, and a Task Tracker. The slave node functions as both a Task Tracker and a Data Node [5].

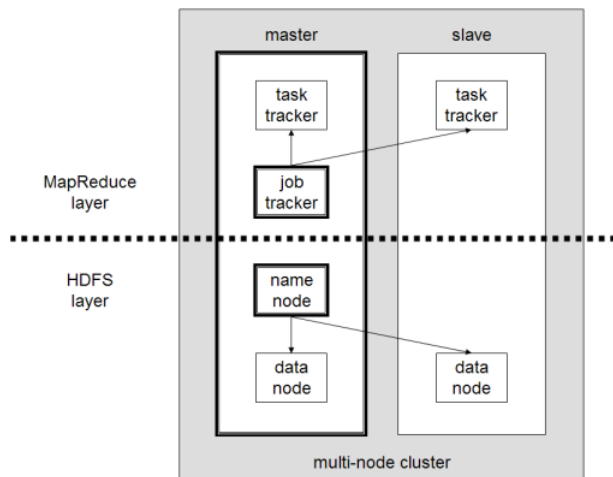


Fig. 2. Hadoop Cluster [4]

Apache Hama

Apache Hama is a top level project in the Apache Software stack developed by Edward J Yoon. Hama utilizes Bulk Synchronous Parallel (BSP) to allow massive scientific computation [6].

Figure 3 details the architecture of Apache Hama. Three key components are BSPMaster, ZooKeeper, and GroomServer. BSPMaster has several responsibilities including maintaining and scheduling jobs and communicating with the groom servers. Groom Servers are processes that perform tasks assigned by BSPMaster. Zookeeper efficiently manages synchronization of BSPPeers, instances started by groom servers [6].

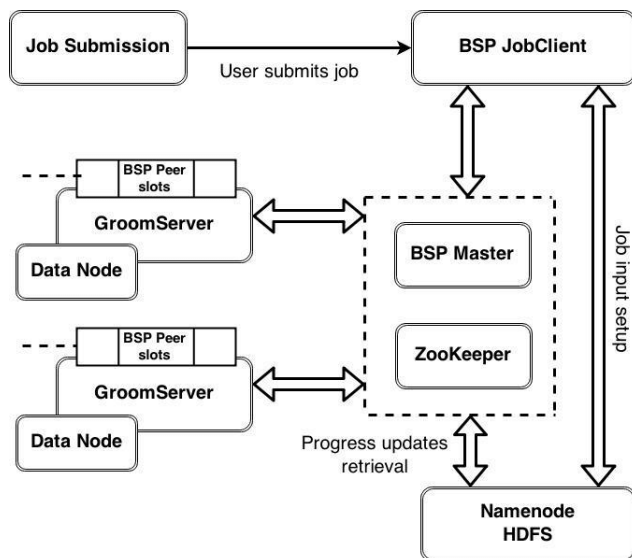


Fig. 3. Hama Architecture [7]

Apache Spark

Apache Spark is open source software, originally developed at the University of California at Berkeley and later donated to the Apache Software Foundation. Spark is a cluster computing framework providing parallelism and fault tolerance [8].

Figure 4 shows the various components of Apache Spark. Spark Core is central to Spark's architecture and provides APIs,

memory management, fault recovery, storage capabilities, and other features. On top of Spark core are several packages. Spark SQL allows the use of SQL for working with structured data. Spark Streaming provides real time streaming capabilities. MLlib and GraphX allow the use of machine learning and graph algorithms [9].

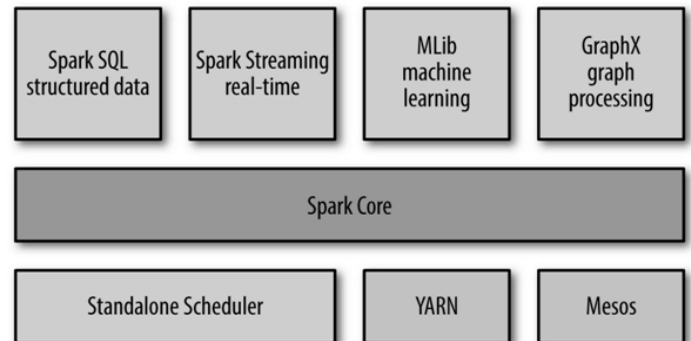


Fig. 4. Spark Architecture [9]

Apache Flink

Apache Flink is open source software developed by the Apache Software Foundation. Flink features a "distributed streaming dataflow engine written in Java and Scala" [10]. Both batch and streaming processing programs can be executed on Flink. Programs in Flink can be written in Java, Scala, Python, and SQL which are compiled and optimized and then executed on a Flink cluster. Flink does not have an internal storage system and instead provides connectors to use with external sources like Amazon S3, Apache Kafka, HDFS, or Apache Cassandra [10]. Flink's full architecture is shown in Figure 5.

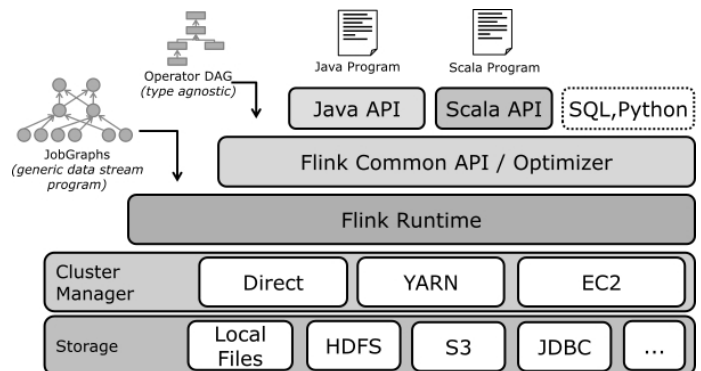


Fig. 5. Flink Architecture [11]

MRQL

MRQL itself is made up of a core module, which includes data formats and structures used by MRQL. Supporting modules, which sit on top of the core module, extend its functionality [12]. MRQL uses a multi-step process to convert SQL statements to Jar files that are then deployed into the cluster. SQL statements are first changed into MRQL algebraic form. Next is a type interface step before the query is translated and normalized. A plan is generated, simplified, normalized, and compiled into java byte code in the final jar file [12].

MAPREDUCE

The MapReduce programming model was introduced by Google in 2004. The MapReduce model involves a map and reduce function. The map function processes key/value pairs to generate a new list of key/value pairs. The reduce function merges these new values by key [13]. MapReduce Query Language is built upon the MapReduce model.

LANGUAGE FEATURES

The MRQL language supports data types, functions, aggregation, and a SQL like syntax.

Data Model

MRQL is a typed language with several data types. Basic types (bool, short, int, long, float, double, string), tuples, records, lists, bags, user-defined types, a data type T, and persistent collections are all supported by MRQL [14].

Data Sources

MRQL can access flat files (such as CSV) and XML and JSON documents [14].

Syntax

MRQL supports a SQL like syntax. MRQL includes group by and order by clauses, nested queries, and three types of repetition [14].

```
select [ distinct ] e
from p1 in e1, ..., pn in en
[ where ec ]
[ group by p': e' [ having eh ] ]
[ order by e0 [ limit e1 ] ]
```

MRQL Select Query Syntax [14]

Functions and Aggregations

In addition to predefined system functions, MRQL supports user defined functions and aggregations [14].

LICENSING

MRQL is open source software licensed under the Apache 2.0 software license [15].

ALTERNATIVE TECHNOLOGIES

There are several existing MapReduce Query Languages that are alternatives to MRQL. Apache Hive (HiveQL), Apache Pig (Pig Latin), and JAQL, a JSON based query language originally developed by Google [16], are three examples. Hive and Pig, popular Apache technologies, are explored in more detail.

HiveQL - Apache Hive

Apache Hive allows the use of the use of SQL (via HiveQL) to access and modify datasets in distributed storage that integrates with Hadoop. Hive also provides a procedural language, HPL-SQL [17].

Table 1 compares key features of MRQL and Hive.

	MRQL	Hive
metadata	none	stored in RDBMS
data	nested collections, trees, custom complex types	relational
group by	on arbitrary queries	not on subqueries
aggregation	arbitrary queries on grouped data	SQL aggregations
platforms	Hadoop, Hama, Spark, Flink	Hadoop, Tez, (Spark)
file formats	text, sequence, XML, JSON	text, sequence, ORC, RCFile
iteration	yes	no
streaming	yes	no

Table 1. MRQL-Hive Comparison [18]

Pig Latin - Apache Pig

Apache Pig was developed at Yahoo in 2006. Like MRQL and Hive, Pig abstracts programming from Java MapReduce to a simpler format. Pig's programming language is called Pig Latin. As opposed to declarative languages where the programmer specifies what will be done like SQL, Hive, and MRQL, Pig Latin is a procedural language where the programmer specifies how the task will be accomplished [19].

Ecosystem

MRQL is part of the vast Apache ecosystem often used when working with Big Data. Other technologies in the Apache Big Data stack closely related to MRQL are Hadoop, Hama, Spark, Flink, and HDFS.

USE CASES

Due to MRQL's relatively recent development and current status in the Apache incubator, real world use cases are difficult to find. Since MRQL can be utilized with Hadoop, Hama, Spark, and Flink, it can be utilized in a wide variety of situations. MRQL can be used for complex data analysis including PageRank, matrix factorization, and k-means clustering [1].

EDUCATIONAL MATERIAL

Several excellent resources exist for learning more about MRQL. The Apache Wiki contains several research papers and presentations on MRQL by its creator Leonidas Fegaras and others [20] which provide a theoretical bases for understanding MRQL. Key resources are *Apache MRQL (incubating): Advanced Query Processing for Complex, Large-Scale Data Analysis* by Leonidas Fegaras [18], *Supporting Bulk Synchronous Parallelism in Map-Reduce Queries* by Leonidas Fegaras [21], and *An Optimization Framework for Map-Reduce Queries* by Leonidas Fegaras, Chengkai Li, and Upa Gupta [22].

The Apache Wiki also contains a *Getting Started* page [23] which describes steps such as downloading and installing

MRQL, a detailed *Language Description* [14], and a listing of *System Functions* [24]. These are the best resources for hands on use of MRQL.

As MRQL is an open source technology, the source code is freely available. It is stored in github [25].

CONCLUSION

MapReduce Query Language simplifies the popular MapReduce programming model frequently utilized with Big Data. MRQL is powerful enough to express complex data analysis including PageRank, matrix factorization, and k-means clustering, yet due to its familiar SQL like syntax can be utilized by a wider variety of users without strong programming skills.

MRQL can be used with Apache Hadoop, Hama, Spark, and Flink. With Hadoop now a mainstream technology and Hama, Spark, and Flink important pieces of the Apache Big Data stack, potential applications of MRQL are wide ranging.

The declarative MRQL language is easy to use, yet powerful, featuring multiple data types, data sources, functions, aggregations, and a SQL like syntax, including order by, group by, nested queries, and repetition.

While currently still an incubator project at Apache, MRQL shows promise as a rich, easy to use language with the flexibility of working with Hadoop, Hama, Spark, and Flink. Due to these strengths, MRQL may emerge as a viable alternative to currently more popular high level MapReduce abstractions such as Hive and Pig.

REFERENCES

- [1] Apache Software Foundation, "Apache incubator," Web Page, accessed 2017-01-29. [Online]. Available: <http://incubator.apache.org/>
- [2] E. J. Yoon, "Mrql - a sql on hadoop miracle," Web Page, Jan. 2017, accessed 2017-01-29. [Online]. Available: <http://www.hadoopsphere.com/2013/04/mrql-sql-on-hadoop-miracle.html>
- [3] Apache Software Foundation, "Apache mrql," Web Page, Apr. 2016, accessed 2017-01-29. [Online]. Available: <https://mrql.incubator.apache.org/>
- [4] Wikipedia, "Apache hadoop," Web Page, Mar. 2017, accessed 2017-03-18. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Hadoop
- [5] E. Coppa, "Hadoop architecture overview," Code Repository, accessed 2017-03-23. [Online]. Available: <http://ercoppa.github.io/HadoopInternals/HadoopArchitectureOverview.html>
- [6] Wikipedia, "Apache hama," Web Page, Dec. 2014, accessed 2017-03-20. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Hama
- [7] K. Siddique, Y. Kim, and Z. Akhtar, "Researching apache hama: A pure bsp computing framework," vol. 2, no. 2. World Academy of Science, Engineering and Technology, 2015, p. 1857. [Online]. Available: <http://waset.org/abstracts/Computer-and-Information-Engineering>
- [8] Wikipedia, "Apache spark," Web Page, Feb. 2017, accessed 2017-03-20. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Spark
- [9] P. Pandey, "Spark programming - rise of spark," Web Page, Aug. 2015, accessed 2017-03-22. [Online]. Available: <http://www.teckstory.com/hadoop-ecosystem/rise-of-spark/>
- [10] Wikipedia, "Apache flink," Web Page, Mar. 2017, accessed 2017-03-20. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Flink
- [11] Apache Software Foundation, "General architecture and process mode," Web Page, accessed 2017-03-18. [Online]. Available: <http://spark.apache.org/docs/1.3.0/cluster-overview.html>
- [12] A. PAUDEL, "Integration of apache mrql query language with apache storm realtime computational system," Master's thesis, The University of Texas at Arlington, Dec. 2016. [Online]. Available: <https://uta-ir.tdl.org/uta-ir/bitstream/handle/10106/26371/PAUDEL-THESIS-2016.pdf?sequence=1>
- [13] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in *Proceedings of the 6th Conference on Symposium on Operating Systems Design & Implementation - Volume 6*, ser. OSDI'04. Berkeley, CA, USA: USENIX Association, 2004, pp. 10–10. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1251254.1251264>
- [14] L. Fegaras, "Mrql: A mapreduce query language," Web Page, Jul. 2014, accessed 2017-03-24. [Online]. Available: <https://wiki.apache.org/mrql/LanguageDescription>
- [15] Apache Software Foundation, "License," Code Repository, Apr. 2013, accessed 2017-03-21. [Online]. Available: <https://github.com/apache/incubator-mrql/blob/master/LICENSE>
- [16] Wikipedia, "Jaql," Web Page, May 2016, accessed 2017-03-20. [Online]. Available: <https://en.wikipedia.org/wiki/Jaql>
- [17] Apache Software Foundation, "Apache hive home," Web Page, Mar. 2017, accessed 2017-03-20. [Online]. Available: <https://cwiki.apache.org/confluence/display/Hive/Home>
- [18] L. Fegaras, "Apache mrql (incubating): Advanced query processing for complex, large-scale data analysis," Web Page, Apr. 2015, accessed 2017-03-14. [Online]. Available: <https://github.com/apache/incubator-mrql/blob/master/LICENSE>
- [19] Wikipedia, "Apache pig (programming tool)," Web Page, Aug. 2016, accessed 2017-03-22. [Online]. Available: [https://en.wikipedia.org/wiki/Pig_\(programming_tool\)](https://en.wikipedia.org/wiki/Pig_(programming_tool))
- [20] L. Fegaras, "Publications," Web Page, Apr. 2015, accessed 2017-03-24. [Online]. Available: <https://wiki.apache.org/mrql/Publications>
- [21] —, "Supporting bulk synchronous parallelism in map-reduce queries," in *Proceedings of the 2012 SC Companion: High Performance Computing, Networking Storage and Analysis*, ser. SCC '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 1068–1077. [Online]. Available: <http://lambda.uta.edu/mrql-bsp.pdf>
- [22] L. Fegaras, C. Li, and U. Gupta, "An optimization framework for map-reduce queries," in *Proceedings of the 15th International Conference on Extending Database Technology*, ser. EDBT '12. New York, NY, USA: ACM, 2012, pp. 26–37. [Online]. Available: <http://lambda.uta.edu/mrql.pdf>
- [23] L. Fegaras, "Getting started with mrql," Web Page, Aug. 2016, accessed 2017-03-24. [Online]. Available: <https://wiki.apache.org/mrql/GettingStarted>
- [24] —, "The mrql system functions," Web Page, Oct. 2016, accessed 2017-03-24. [Online]. Available: <https://wiki.apache.org/mrql/SystemFunctions>
- [25] Apache Software Foundation, "Apache mrql," Code Repository, accessed 2017-03-25. [Online]. Available: <https://git-wip-us.apache.org/repos/asf?p=incubator-mrql.git>

AUTHOR BIOGRAPHY

Mark McCombe received his B.S. (Business Administration/Finance) and M.S. (Computer Information Systems) from Boston University. He is currently studying Data Science at Indiana University Bloomington.

WORK BREAKDOWN

All work on this paper was completed solely by Mark McCombe.