Big Data Software

Spring 2017

Bloomington, Indiana

Editor:
Gregor von Laszewski
Department of Intelligent Systems
Engeneering
Indiana University
laszewski@gmail.com

Contents

Berkeley DB

SABER SHEYBANI[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: sheybani@indiana.edu*

Paper 2, April 30, 2017

Berkeley DB is a family of open source, NoSQL key-value database libraries. It provides a simple function-call API for data access and management over a number of programming languages, including C, C++, Java, Perl, Tcl, Python, and PHP. Berkeley DB is embedded because it links directly into the application and runs in the same address space as the application. As a result, no inter-process communication, either over the network or between processes on the same machine, is required for database operations. It is also extremely portable and scalable, it can manage databases up to 256 terabytes in size. For data management, Berkeley DB offers advanced services, such as concurrency for many users, ACID transactions, and recovery. Berkeley DB is used in a wide variety of products and a large number of projects, including gateways from Cisco, Web applications at Amazon.com and open-source projects such as Apache and Linux.

Keywords: NoSQL, embedded database, Oracle, open source database management system

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-ER-1001/report.pdf

1. INTRODUCTION

Data management has always been a fundamental issue in programming. Since 1960s, countless database management systems have been developed to fulfil different sorts of demands. The question for every user is choosing the system that best fits the requirements of its appliction.

Database management systems can be categorized based on data models, into a number of groups: Hierarchical Databases, Network Databases, Relational Databases, Object-based Databases, and Semistructured Databases.
Hierarchical databases use the oldest type of data models, which is a tree-like structure. The records are connected to each other with a hard-coded link.
In a **Network** model, records are also connected with links, but there is no hierarchy. Instead, the structure is graph-like and all of the nodes can connect to each other.
In **Relational** databases, there are no physical links, but the data is structured in tables (relations). Each row represents a record and each column represents an attribute. The tables are connected with common attributes, which makes querying much easier than the two former models. For this reason, database management systems using relational models are the most widely used ones.
Object-based data models extend concepts of object-oriented programming into database systems, in order to provide persistent storage of objects and other capabilities of databases for object-oriented programming.

Semistructured databases which include NoSQL databases are the type of database model that enable storage of hetereogenous data, by allowing records with different attributes. This however, is achieved by sacrificing the knowledge of data type by the database system. The data in this case must be *self-describing*, meaning that the description (schema) of the data must be in itself. XML (Extensible Markup Language) schema language is a widely used language for providing schema for these database systems[1].

Berkeley DB fits into the last category, as a NoSQL database system. The records are stored as key-value pairs and a few logical operations can be executed on them, namely: insertion, deletion, finding a record by its key, and updating an already found record. "Berkeley DB never operates on the value part of a record. Values are simply payload, to be stored with keys and reliably delivered back to the application on demand." There is no notion of schema and no support for SQL queries. "The application must understand the keys and values that it uses. On the other hand, there is literally no limit to the data types that can be stored in a Berkeley DB database. The application never needs to convert its own program data into the data types that Berkeley DB supports. Berkeley DB is able to operate on any data type the application uses, no matter how complex" [2].

2. ARCHITECTURE

Berkeley DB's architecture can be explained by five major subsystems: **Access Methods**: Providing general-purpose support

for creating and accessing database files. **Memory Pool**: The general-purpose shared memory buffer pool. Multiple **Transaction**: Implementing the transaction model, realizing ACID properties. processes and threads within processes share access to databases using this subsystem. **Locking**: The general-purpose lock manager for processes. **Logging**: The write-ahead logging that supports the Berkeley DB transaction model.

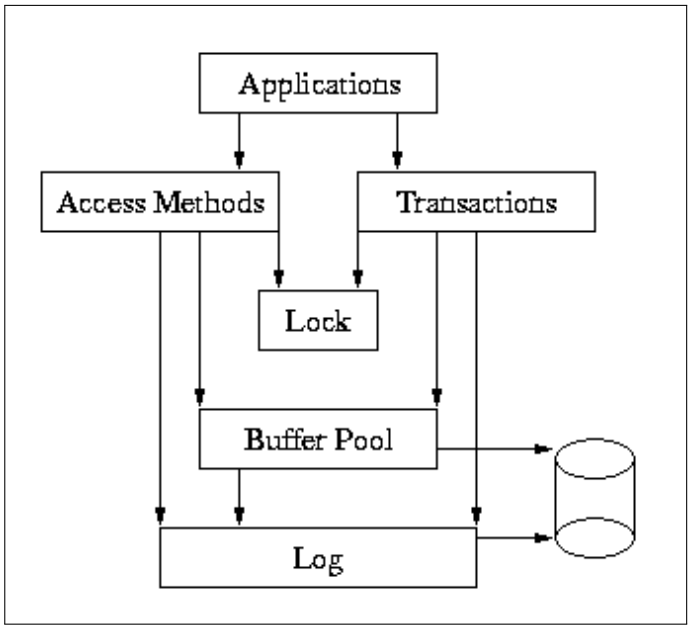


Fig. 1. Berkeley DB Subsystems [3]

Figure 1 displays a diagram of the Berkeley DB library architecture. The arrows are calls that invoke the destination. Each subsystem can also be used independent from the other ones, but this usage is not common.

3. SERVICES AND OTHER FEATURES

The two fundamental services that every database management system provides are data access, and data management services. Data access services include the low-level operations on the records, which were already mentioned in the introduction for the case of Berkeley DB. In terms of storage structure, Berkeley DB supports **hash tables, Btrees, simple record-number-based storage, and persistent queues** [4].

Data management services are the higher-level services (and features) such as concurrency that ensure specific qualities for operation of the system. These services include allowing simultaneous access to the records by multiple users (concurrency), changing multiple records at the same time (transaction), and complete recovery of the data from crashes (recovery) [4].
For **concurrency**, Berkeley DB is able to handle low-level services such as locking and shared buffer management transparently, while multiple processes and threads use the data.
For recovery, every application can ask Berkeley DB for recovery, at startup time.
An **ACID transaction** ensures the following specifications at the end of its operation [5]: Atomicity (Either all or none of the records change), Consistency (The system goes from one valid state to another), Isolation (concurrent execution of multiple transaction yields the same result as the sequential execution of them), Durability (The result remains steady, even in case of crash of the system).Berkeley DB "libraries provide strict ACID transaction semantics, by default. However, applications are allowed to relax the isolation guarantees the database system makes" [4].

Berkeley DB runs in the same address space as the application. As a result, there is no need for communication between processes and threads. On the other hand, as an embedded database management system, it does not provide a standalone server. However, server applications can be built over Berkeley DB and many examples of Lightweight Directory Access Protocol (LDAP) servers have been built using it [2].

The database library for Berkeley DB consumes less than 300 kilobytes of text space on common architectures. That makes it a feasible solution for embedded systems with small capacities. Nonetheless, it can manage up to 256 terabytes databases.

3.1. Supported Operating Systems and Languages

Berkeley DB supports nearly all modern operating systems. They include Windows, Linux, Mac OS X, Android, iPhone, Solaris, BSD, HP-UX, AIX, and RTOS such as VxWorks,and QNX. The supported programming languages include "C, C++, Java, C#, Perl, Python, PHP, Tcl, Ruby and many others" [6].

3.2. Required Infrastructure

As infrastructure, Berkeley DB requires "underlying IEEE/ANSI Std 1003.1 (POSIX) system calls and can be ported easily to new architectures by adding stub routines to connect the native system interfaces to the Berkeley DB POSIX-style system calls" [7].

4. PRODUCTS AND LICENSING

The products include three implementations on C, C++, and Java (Oracle Berkeley DB, Oracle Berkeley XML, and Oracle Berkeley JE, respectively) [8].
Berkeley DB is an open source library and is free for use and redistribution in other open source products. the distribution includes complete source code for all three implementations, their supporting utilities, as well as complete documentation in HTML format [7].

For redistribution in commercial products, Sleepycat Software licenses four products, with prices ranging from US$900 to 13,800 per processor [9] as of March 2017. The products, in the order of ascending price and capabilities are: Berkeley DB Data Store, Berkeley DB Concurrent Data Store, Berkeley DB Transactional Data Store, Berkeley DB High Availability. The Sleepycat software also includes prebuilt libraries and binaries as part of support services, which is not provided in the free distribution. There is no additional license payment for embedded usage within the Oracle Retail Predictive Application Server (RPAS).

5. USE CASES

A notable number of open source and commercial products in different areas of technology, use Berkeley DB. Open source use cases include Linux, UNIX, BSD, Apache, Solaris, MySQL, Sendmail, OpenLDAP, and MemcacheDB.
Proprietary applications "include directory servers from Sun and Hitachi; messaging servers from Openwave and LogicaCMG; switches, routers and gateways from Cisco, Motorola, Lucent, and Alcatel; storage products from EMC and HP; security products from RSA Security and Symantec; and Web applications at Amazon.com, LinkedIn and AOL" [6].

6. ADVANTAGES AND LIMITATIONS

Berkeley DB has two advantages over relational and object-oriented database systems, when it comes to embedded applications. One is running in the same address space as the application and thus, not requiring any inter-process communication which can have a high cost in embedded applications. And the other is simplicity of interface for operations which does not require query language parsing. These two features along with its small size, give Berkeley DB system a privilege of being lightweight enough for many applications where there are tight constraints on resources.
However, with simplicity comes the lack of SQL features. If the user of the application needs to perform complicated searches (potentially using SQL queries) the programmer would need to write the code for those cases. In general, Berkeley DB is aimed at providing fast, reliable, transaction-protected record storage, at a minimalist way [10].

7. EDUCATIONAL MATERIAL

As was mentioned in the Products section, the free distribution comes with complete documentation in HTML format. The documentation has two parts: a reference manual in UNIX-style for programmers, and a reference guide which can serve as a tutorial [11]. In addition to that, *Berkeley DB Tutorial and Reference Guide, Version 4.1.24* [7] and *The Berkeley DB Book* [12] are useful resources for learning more about Berkeley DB and getting started with it.

8. CONCLUSION

Berkeley DB is a minimal, lightweight database management system, focused on providing performance, especially in embedded systems. It offers a small, simple set of data access services, and a rich powerful set of data management services. It is freely available for use by non-commercial distributions and has been successfully used in many projects.

REFERENCES

[1] I. Limited, *Introduction to Database Systems:*. Pearson, 2010. [Online]. Available: https://books.google.com/books?id=-YY-BAAAQBAJ

[2] "What berkeley db is not," Web Page, accessed: 2017-4-6. [Online]. Available: https://web.stanford.edu/class/cs276a/projects/docs/berkeleydb/ref/intro/dbisnot.html

[3] "The big picture," Web Page, accessed: 2017-4-6. [Online]. Available: https://web.stanford.edu/class/cs276a/projects/docs/berkeleydb/ref/arch/bigpic.html

[4] "What is berkeley db?" Web Page, accessed: 2017-4-6. [Online]. Available: https://web.stanford.edu/class/cs276a/projects/docs/berkeleydb/ref/intro/dbis.html

[5] T. Haerder and A. Reuter, "Principles of transaction-oriented database recovery," *ACM Computing Surveys (CSUR)*, vol. 15, no. 4, pp. 287–317, 1983.

[6] "Oracle berkeley database products," Web Page, accessed: 2017-4-6. [Online]. Available: http://www.oracle.com/technetwork/products/berkeleydb/learnmore/berkeley-db-family-datasheet-132751.pdf

[7] "Berkeley db tutorial and reference guide, version 4.1.24," Web Page, accessed: 2017-4-6. [Online]. Available: https://web.stanford.edu/class/cs276a/projects/docs/berkeleydb/reftoc.html

[8] "Oracle berkeley db 12c," Web Page, accessed: 2017-4-6. [Online]. Available: http://www.oracle.com/technetwork/database/database-technologies/berkeleydb/%20overview/index.html

[9] "Oracle technology global price list," Web Page, accessed: 2017-4-6. [Online]. Available: http://www.oracle.com/us/corporate/pricing/technology-price-list-070617.pdf

[10] "Do you need berkeley db?" Web Page, accessed: 2017-4-6. [Online]. Available: https://web.stanford.edu/class/cs276a/projects/docs/berkeleydb/ref/intro/need.html

[11] M. A. Olson, K. Bostic, and M. I. Seltzer, "Berkeley db." in *USENIX Annual Technical Conference, FREENIX Track*, 1999, pp. 183–191.

[12] H. Yadava, *The Berkeley DB Book*, ser. Books for Professionals by Professionals. Apress, 2007. [Online]. Available: https://books.google.com/books?id=2wEkW7pQ0KwC

AUTHOR BIOGRAPHY

Saber Sheybani received his B.S. (Electrical Engineering - Minor in Control Engineering) from University of Tehran. He is currently a PhD student of Intelligent Systems Engineering - Neuroengineering at Indiana University Bloomington.

Apache Ranger

AVADHOOT AGASTI[1,*,+]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: aagasti@indiana.edu
[+] HID - SL-IO-3000

paper2, April 30, 2017

Apache Hadoop provides various data storage, data access and data processing services. Apache Ranger is part of the Hadoop ecosystem. Apache Ranger provides capability to perform security administration tasks for storage, access and processing of data in Hadoop. Using Ranger, Hadoop administrator can perform security administration tasks using a central user interface or restful web services. Hadoop administrator can define policies which enable users or user-groups to perform specific actions using Hadoop components and tools. Ranger provides role based access control for datasets on Hadoop at column and row level. Ranger also provides centralized auditing of user access and security related administrative actions.

Keywords: Apache Ranger, LDAP, Active Directory, Apache Knox, Apache Atlas, Apache Hive, Apache Hadoop, Yarn, Apache HBase, Apache Storm, Apache Kafka, Data Lake, Apache Sentry, Hive Server2, Java

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IO-3000/report.pdf

1. INTRODUCTION

Apache Ranger is open source software project designed to provide centralized security services to various components of Apache Hadoop. Apache Hadoop provides various mechanism to store, process and access the data. Each Apache tool has its own security mechanism. This increases administrative overhead and is also error prone. Apache Ranger fills this gap to provide a central security and auditing mechanism for various Hadoop components. Using Ranger, Hadoop administrator can perform security administration tasks using a central user interface or restful web services. The administrator can define policies which, enable users or user-groups to perform specific actions using Hadoop components and tools. Ranger provides role based access control for datasets on Hadoop at column and row level. Ranger also provides centralized auditing of user access and security related administrative actions.

2. ARCHITECTURE OVERVIEW

[1] describes the important components of Ranger as explained below:

2.1. Ranger Admin Portal

Ranger admin portal is the main interaction point for the user. A user can define policies using the Ranger admin portal. These policies are stored in a policy database. The Policies are polled by various plugins. Admin portal also collects the audit data from plugins and stores it in HDFS or in a relational database.

2.2. Ranger Plugins

Plugins are Java programs, which are invoked as part of the cluster component. For example, the ranger-hive plugin is embedded as part of Hive Server2. The plugins cache the policies, and intercept the user request and evaluates it against the policies. Plugins also collect the audit data for that specific component and send to admin portal.

2.3. User group sync

While Ranger provides authorization or access control mechanism, it needs to know the users and the groups. Ranger integrates with unix user management system or LDAP or active directory to fetch the users and the groups information. The user group sync component is responsible for this integration.

3. HADOOP COMPONENTS SUPPORTED BY RANGER

Ranger supports auditing and authorization for following Hadoop components [2].

3.1. Apache Hadoop and HDFS

Apache Ranger provides plugin for Hadoop, which helps in enforcing data access policies. The HDFS plugin works with name node to check if the user's access request to a file on HDFS is valid or not.

3.2. Apache Hive

Apache Hive provides SQL interface on top of the data stored in HDFS. Apache Hive supports two types of authorization:

storage based authorization and SQL standard authorization. Ranger provides centralized authorization interface for Hive, which provides granular access control at table and column level. Ranger's hive plugin is part of Hive Server2.

3.3. Apache HBase

Apache HBase is NoSQL database implemented on top of Hadoop and HDFS. Ranger provides coprocessor plugin for HBase, which performs authorization checks and audit log collections.

3.4. Apache Storm

Ranger provides plugin to Nimbus server which helps in performing the security authorization on Apache Storm.

3.5. Apache Knox

Apache Knox provides service level authorization for users and groups. Ranger provides plugin for Knox using which, administration of policies can be supported. The audit over Knox data enables user to perform detailed analysis of who and when accessed Knox.

3.6. Apache Solr

Solr provides free text search capabilities on top of Hadoop. Ranger is useful to protect Solr collections from unauthorized usage.

3.7. Apache Kafka

Ranger can manager access control on Kafka topics. Policies can be implemented to control which users can write to a Kafka topic and which users can read from a Kafka topic.

3.8. Yarn

Yarn is resource management layer for Hadoop. Administrators can setup queues in Yarn and then allocate users and resources per queue basis. Policies can be defined in Ranger to define who can write to various Yarn queues.

4. IMPORTANT FEATURES OF RANGER

The blog article [3] explains the 2 important features of Apache Ranger.

4.1. Dynamic Column Masking

Dynamic data masking at column level is an important feature of Apache Ranger . Using this feature, the administrator can setup data masking policy. The data masking makes sure that only authorized users can see the actual data while other users will see the masked data. Since the masked data is format preserving, they can continue their work without getting access to the actual sensitive data. For example, the application developers can use masked data to develop the application whereas when the application is actually deployed, it will show actual data to the authorized user. Similarly, a security administrator may chose to mask credit card number when it is displayed to a service agent.

4.2. Row Level Filtering

The data authorization is typically required at column level as well as at row level. For example, in an organization which is geographically distributed in many locations, the security administrator may want to give access of a data from a specific location to the specific user. In other example, a hospital data security administrator may want to allow doctors to see only his or her patients. Using Ranger, such row level access control can be specified and implemented.

5. HADOOP DISTRIBUTION SUPPORT

Ranger can be deployed on top of Apache Hadoop. [4] provides detailed steps of building and deploying Ranger on top of Apache Hadoop.

Hortonwork Distribution of Hadoop(HDP) supports Ranger deployment using Ambari. [5] provides installation, deployment and configuration steps for Ranger as part of HDP deployment.

Cloudera Hadoop Distribution (CDH) does not support Ranger. According to [6], Ranger is not recommended on CDH and instead Apache Sentry should be used as central security and audit tool on top of CDH.

6. USE CASES

Apache Ranger provides centralized security framework which can be useful in many use cases as explained below.

6.1. Data Lake

[7] explains that storing many types of data in the same repository is one of the most important feature of data lake. With multiple datasets, the ownership, security and access control of the data becomes primary concern. Using Apache Ranger, the security administrator can define fine grain control on the data access.

6.2. Multi-tenant Deployment of Hadoop

Hadoop provides ability to store and process data from multiple tenants. The security framework provided by Apache Ranger can be utilized to protect the data and resources from un-authorized access.

7. APACHE RANGER AND APACHE SENTRY

According to [8], Apache Sentry and Apache Ranger have many features in common. Apache Sentry ([9]) provides role based authorization to data and metadata stored in Hadoop.

8. EDUCATIONAL MATERIAL

[10] provides tutorial on topics like A)Security resources B)Auditing C)Securing HDFS, Hive and HBase with Knox and Ranger D) Using Apache Atlas' Tag based policies with Ranger. [11] provides step by step guidance on getting latest code base of Apache Ranger, building and deploying it.

9. LICENSING

Apache Ranger is available under Apache 2.0 License.

10. CONCLUSION

Apache Ranger is useful to Hadoop Security Administrators since it enables the granular authorization and access control. It also provides central security framework to different data storage and access mechanism like Hive, HBase and Storm. Apache Ranger also provides audit mechanism. With Apache Ranger, the security can be enhanced for complex Hadoop use cases like Data Lake.

ACKNOWLEDGEMENTS

The authors thank Prof. Gregor von Laszewski for his technical guidance.

REFERENCES

[1] Hortonworks, “Apache ranger - overview,” Web Page, online; accessed 9-Mar-2017. [Online]. Available: https://hortonworks.com/apache/ranger/#section_2

[2] A. S. Foundation, “Apache ranger - frequenty asked questions,” Web Page, online; accessed 9-Mar-2017. [Online]. Available: http://ranger.apache.org/faq.html#How_does_it_work_over_Hadoop_and_related_components

[3] S. Mahmood and S. Venkat, “For your eyes only: Dynamic column masking & row-level filtering in hdp2.5,” Web Page, Sep. 2016, online; accessed 9-Mar-2017. [Online]. Available: https://hortonworks.com/blog/eyes-dynamic-column-masking-row-level-filtering-hdp2-5/

[4] A. S. Foundation, “Apache ranger 0.5.0 installation,” Web Page, online; accessed 9-Mar-2017. [Online]. Available: https://cwiki.apache.org/confluence/display/RANGER/Apache+Ranger+0.5.0+Installation

[5] Horto, “Installing apache rang,” Web Page, online; accessed 9-Mar-2017. [Online]. Available: https://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.3.6/bk_installing_manually_book/content/ch_installing_ranger_chapter.html

[6] Cloudera, “Configuring authorization,” Web Page, online; accessed 9-Mar-2017. [Online]. Available: https://www.cloudera.com/documentation/enterprise/5-6-x/topics/sg_authorization.html

[7] Teradata and Hortonworks, “Putting the data lake to work - a guide to best practices,” Web Page, Apr. 2014, online; accessed 9-Mar-2017. [Online]. Available: https://hortonworks.com/wp-content/uploads/2014/05/TeradataHortonworks_Datalake_White-Paper_20140410.pdf

[8] S. Neumann, “5 hadoop security projects,” Web Page, Nov. 2014, online; accessed 9-Mar-2017. [Online]. Available: https://www.xplenty.com/blog/2014/11/5-hadoop-security-projects/

[9] A. S. Foundation, “Apache sentry,” Web Page, online; accessed 9-Mar-2017. [Online]. Available: https://sentry.apache.org/

[10] Hortonworks, “Apache ranger overview,” Web, online; accessed 9-Mar-2017. [Online]. Available: https://hortonworks.com/apache/ranger/#tutorials

[11] A. S. Foundation, “Apache ranger - quick start guide,” Web Page, online; accessed 9-Mar-2017. [Online]. Available: http://ranger.apache.org/quick_start_guide.html

Amazon Kinesis

ABHISHEK GUPTA[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: abhigupt@iu.edu*

project-001, April 30, 2017

Amazon Kinesis [1] provides a software-as-a-service(SAAS) platform for application developers working on Amazon Web Services(AWS) [2] platform. Kinesis is capable of processing streaming data at in real time. This is a key challenge application developers face when they have to process huge amounts of data in real time. It can scale up or scale down based on data needs of the system. As volume of data grows with advent IOT [3] devices and sensors, Kinesis will play a key role in developing applications which require insights in real time with this growing volume of data.

Keywords: Cloud, I524

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IO-3005/report.pdf

1. INTRODUCTION

Amazon Kinesis [1] helps application developers collect and analyze streaming data in real time. The streaming data can come from variety of sources like social media, sensors, mobile devices, syslogs, logs, web server logs, network data etc. Kinesis can scale on demand as application needs change. For example during peak load situation kinesis added more workers nodes and can reduce the nodes when the application runs at low load. It also provides durability, where if one of the node goes down the data is persisted on disk and get replicated when new nodes come up. Multiple applications can consume data from one or more streams for variety of use cases for example, one application computes moving average and another application counts the number of users clicks. These applications can work in parallel and independently. Kinesis provides streaming in realtime with sub-second delays between producer and consumer. Kinesis has two types of processing engines: Kinesis streams - reads data from producers and Kinesis firehose - pushes data to consumers.

Kinesis streams can be used to process incoming data from multiple sources. Kinesis firehose is used to load streaming data into AWS like Kinesis analytics, S3 [4], Redshift [5], Elasticsearch [6] etc.

2. ARCHITECTURE

2.1. Introduction

Amazon Kinesis reads data from variety of sources. The data coming into streams is in a record format. Each record is composed on a partition key, sequence number and data blob which is raw serialized byte array. The data further is partitioned into multiple shards(or workers) using the partition key.

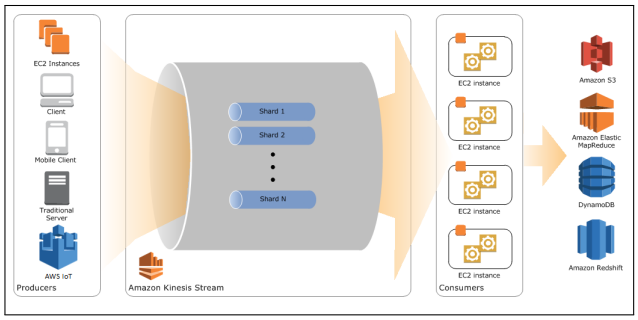


Fig. 1. Kinesis streams building blocks [7]

2.2. Building blocks

Following are key components in streams architecture [7] :

2.2.1. Data Record

Its one unit of data that flows through Kinesis stream. Data records is made up of sequence number, partition key, and blob of actual data. Size of data blob is max 1 MB. During aggregation one or more records are aggregated in to a single aggregated record. Further these aggregated records are emitted as an aggregation collection.

2.2.2. Producer

Producers write the data to Kinesis stream. Producer can be any system producing data. For example, ad server, social media stream, log server etc.

2.2.3. Consumer

Consumers subscribe to one or more streams. Consumer can be one of the applications running on AWS or hosted on EC2[8]

Fig. 2. Aggregation of records

instance(virtual machines).

2.2.4. Shard

A shard is an instance of kinesis stream engine. A stream can have one or more shards. Records are processed by each shard based on the partition key. Each shard can process up to 2MB/s data for reads and up to 1MB/s for writes. Total capacity of a stream is sum of capacities of its shards.

2.2.5. Partition Key

Partition key is 256 bytes long. A MD5 [9] hash function is used to map partition keys to 128 bit integer value which is further used to map to appropriate shard.

2.2.6. Sequence Number

Sequence number is assigned to a record when a record get written to the stream.

2.2.7. Amazon Kinesis Client Library

Amazon Kinesis Client Library is bundled into your application built on AWS. It makes sure that for each record there is a shard available to process that record. Client library uses dynamo db to store control data records being processed by shards.

2.2.8. Application Name

Name of application is stored in the control table in DynamoDB [10] where kinesis streams will write the data to. This name is unique.

3. KINESIS DEVELOPMENT

AWS provides a java SDK. Java SDK [11] can be used to complete all workflows on stream. Workflow like create, listing, retrieving shards from stream, deleting stream, re-sharding stream and changing data retention period. SDK provide rich documentation and developer blogs to support development on streams.

You can create and deploy Kinesis components using following: Kinesis console, Streams API, and AWS CLI.

Before creating stream you should determine initial size of the stream [12] and number of shards required to create your stream. Number of shards can be calculated using the formulae:

$$NumberOfShards = max(\frac{A}{1000}, \frac{B}{2000})$$

A = Incoming Write Bandwidth In KB
B = Outgoing Read Bandwidth In KB

Here, the attributes used in the calculation are self explanatory. Producer for streams writes data records into Kinesis streams. This data is available for 24 hours within streams. The default retention interval can be changed. To write records to stream, you must specify partition key, name of stream and data blob. Consumer on the other hand read data from streams using shard iterator. Shard iterator provides consumer a position on streams from where the consumer can start reading the data.

4. STREAM LIMITS

4.1. Shard

Kinesis streams has certain limits [13] : by default there can 25 shards in a region except US east, EU and US west have limit of 50 shards. Each shard can support up to 5 transactions per second for reads and at maximum data rate of 2 MB per second. Each shard can support 1000 records per second for writes and maximum data rate of 1MB per second.

4.2. Data retention

By default the data is available for 24 hours which can be configured up to 168 hours with 1 hour increments.

4.3. Data Blob

Maximum size of data blob is 1MB before base64 encoding [13].

5. MANAGEMENT

Kinesis provides all management using AWS console or you can build a custom management application using Java SDK [11] provided by AWS. AWS provides a web console to manage all AWS services including kinesis. Using console web user interface user can perform all operations to manage stream.

6. MONITORING

AWS provides several ways to monitor its services. Kinesis can use these services for monitoring purpose: CloudWatch metrics, Kinesis Agent, API logging, Client library, and Producer Library

CloudWatch metrics allows you can monitor the data and usage at shard level. It can collect metrics like: latency, incoming bytes, incoming records, success count etc.

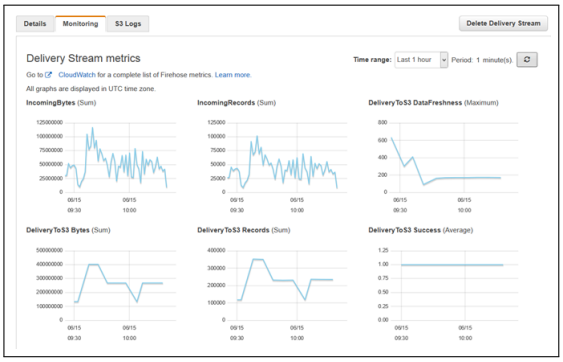


Fig. 3. Kinesis Metrics[14]

7. LICENSING

Kinesis is software as a service(SAAS) from Amazon AWS infrastructure. Hence it can only run as a service within AWS. It comes with pay-as-you-go pricing.

8. USE CASES

Kinesis streams [1] and firehose can be useful in variety of use cases: log data processing, log mining,realtime metrics, reporting realtime analytics, and complex stream processing

For example, Kinesis can be used in serving advertisements based on user click events. Where clients send the clickstream data to Kinesis streams. Stream further generates records which are processed by spark streaming. Further, the algorithms running on spark streaming can be used to generate insights based on user's interest.

> We send clickstream data containing content and audience information from 250+ digital properties to Kinesis Streams to feed our real-time content recommendations engine so we can maximize audience engagement on our sites - Hearst Corporation [1]

Kinesis solves variety of these business problems by doing a real time analysis and aggregation. This aggregated data can further be stored or available to query. Since it runs on amazon, it becomes easy for users to integrate and use other AWS components.

9. CONCLUSION

Kinesis can process huge amounts of data in realtime. Application developers can then focus on business logic. Kinesis can help build realtime dashboards, capture anomalies, generate alerts, provide recommendations which can help take business and operation decisions in real time. It can also send data to other AWS services. You can scale up or scale down as application demand increases or decreases and only pay based on your usage. Only downside of Kinesis it that it cannot run on a private or hybrid cloud, rather can only run on AWS public cloud or Amazon VPC(Virtual Private Cloud)[15]. Customers who want to use Kinesis but don't want to be on Amazon platform cannot use it.

ACKNOWLEDGEMENTS

Special thanks to Professor Gregor von Laszewski, Dimitar Nikolov and all associate instructors for all help and guidance related to latex and bibtex, scripts for building the project, quick and timely resolution to any technical issues faced. The paper is written during the course I524: Big Data and Open Source Software Projects, Spring 2017 at Indiana University Bloomington.

REFERENCES

[1] "Kinesis - real-time streaming data in the aws cloud," Web Page, accessed: 2017-01-17. [Online]. Available: https://aws.amazon.com/kinesis/

[2] "AWS - amazon web services," Web Page, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/

[3] K. Hwang, J. Dongarra, and G. C. Fox, *Distributed and cloud computing: from parallel processing to the internet of things*, T. Green and R. Day, Eds. Morgan Kaufmann, 2012. [Online]. Available: https://www.amazon.com/Distributed-Cloud-Computing-Parallel-Processing-ebook/dp/B00GNBLGE4?SubscriptionId=0JYN1NVW651KCA56C102&tag=techkie-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=B00GNBLGE4

[4] "Amazon S3 simple durable, massively scalable object storage," Web Page, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/s3

[5] "Amazon Redshift fats, simple, cost-effective data warehousing," Web Page, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/redshift/

[6] "Amazon Elasticsearch fully managed, reliable, and scalable elasticsearch service," Web Page, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/elasticsearch-service

[7] "Amazon Kinesis streams key concepts," Web Page, accessed: 2017-03-15. [Online]. Available: http://docs.aws.amazon.com/streams/latest/dev/key-concepts.html

[8] "Amazon EC2 secure and resizable compute capacity in cloud," Web Page, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/ec2/

[9] R. Rivest, "Rfc 1321," *The MD-5 Message Digest Algorithm, SRI Network Information Center*, no. 1321, Apr. 1992. [Online]. Available: https://tools.ietf.org/html/rfc1321

[10] "Amazon S3 fast and flexible NoSQL database," Web Page, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/dynamodb/

[11] "Kinesis - aws sdk for java," Web Page, accessed: 2017-03-15. [Online]. Available: https://aws.amazon.com/sdk-for-java/

[12] "Kinesis - real-time streaming data in the aws cloud," Web Page, accessed: 2017-03-15. [Online]. Available: http://docs.aws.amazon.com/streams/latest/dev/amazon-kinesis-streams.html

[13] "Amazon Kinesis streams limits," Web Page, accessed: 2017-04-01. [Online]. Available: http://docs.aws.amazon.com/streams/latest/dev/service-sizes-and-limits.html

[14] B. Liston, "Serverless cross account stream replication using aws lambda, amazon dynamodb, and amazon kinesis firehose," Web Page, Aug. 2016, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/blogs/compute/tag/amazon-kinesis/

[15] "Amazon virtual private cloud (VPC)," Web Page, accessed: 2017-04-01. [Online]. Available: https://aws.amazon.com/vpc/

Google Cloud DNS

VISHWANATH KODRE[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: vkodre@iu.edu

paper2, April 25, 2017

Google CloudPlatform is one of the key player in providing cloud based services (IaaS, PaaS and SaaS) and solutions to their customers. The users of the Google cloud platform have the flexibility to custom their services as per their requirement fitting to their application architecture or design and budget. Google has established their CloudPlatform infrastructure worldwide. With the high availability, Google Cloud Platform offers their users NO or very is very less downtime, low latency and high throughput.
Google Cloud DNS is fairly new service added by google with an aim to lower the latency of the application or the website loading time, as most of the complicated websites now a days has resources referencing to multiple/different DNS addresses and resolution of such DNS by application per user of the website takes lot of time and slows down the rendering of the website to their end user. With an introduction to Cloud DNS Google's customer can improve the speed of loading the site as has multiple things/services to offer their customers such as zonal distribution of the DNS, caching and programmable feature/customization which offers Google's Cloud DNS users an opportunity to enhance experience of the web app.
Though Google Cloud DNS being new entry in already established Cloud DSN market, Google is giving though competition to other providers such as Amazon Route 53. Google Cloud DNS has lot to offer to their user as it not only a very reliable DNS service, it leverages the cloud infrastructure which Google has established, also Google Cloud DNS comes with very affordable pricing.

Keywords: Google Cloud Platform and Cloud DNS, I524

https://github.com/cloudmesh/sp17-i524/tree/master/paper1/S17-IO-3008/report.pdf

1. IMPORTANCE OF DNS

DSN maintains the mapping between the name and actual IP address of the website. Many websites has complex pages which maps or needs to resolve from multiple IP address. Resolving these names and corresponding IP address decides the uptime or response time of the website along with infrastructure web site is deployed on and other deciding factors of the speed of the website. DNS resolution is one of the key deciding factor to decide the response time of the website. Google's Cloud DNS is a pure cloud-based DNS service, which don't handle domain registration but offers higher control and more features on the service itself.

2. GOOGLE PUBLIC DNS AND CLOUD DNS

Google offers services of caching resolved DNS to overcome the performance challenges while resolving public DNS that complex web pages includes resources from multiple origin domains, which leads to performance hit due to multiple lookups, "Google Public DNS is a recursive DNS resolver, similar to other publicly available services"[1] In comparison Google Cloud DNS is designed for very high volume, programable, autoreactive name server which leverages the power of google cloud infrastructure and it just add up more performance, security, correctness which public DNS already are offering, it guaranties the high availability and distribution of the DNS service per zone which add ups the performance boost while resolving multiple domains.

3. TECHNOLOGIES PROVIDED

The concept of google cloud DNS is built around "Projects, Managed Zones, Record sets and changes to record sets"[2]

3.1. Project

A Platform for managing the projects, resources, domain access control and place billing is configured, every cloud DNS resource lives within project and every cloud DNS operation must specify the project to work with.

3.2. Managed Zones

A managed zone is collection or metadata info of DNS zones which holds the records for same DNS suffix. Inside a project there could be multiple managed zones identified by unique name. "They are automatically assigned named server when they are created to handle responding to DNS queries for that zone." [3]

To create the managed zone for enabling cloud DNS with google one has to create google account and enable the cloud DNS service on their account followed by choosing the compute engine or by creating new Project. After completing the prerequisites user can create the new zones or delete existing ones.

To create a zone user have to provide DNS zone name, description and name to identify zone. Newly created zone is connected to google cloud DNS project automatically however it is not in use until user update their domain registration or explicitly point some resolver at or directly query zone's name servers.

e.g. for creating new zone "gcloud dns managed-zones create –dns-name="example.com." –description="A zone" "myzonename"[2]

3.3. Resource record sets collection

"The resource record sets collection holds the current state of the DNS records that make up a managed zone." [2] It is read only resource collection which can be modified from manage zone creating Change request in the changes collection and it reflects immediately. "User can easily send the desired changes to the API using the import, export, and transaction commands" [2] Importing record set into Managed Zone the format can be of BIND zone file format or YAML record format.

To import user need to run "dns record-set import" e.g. "gcloud dns record-sets import -z=examplezonename –zone-file-format path-to-example-zone-file" [2]

To Export user need to run "dns record-sets export" e.g. "gcloud dns record-sets export -z=examplezonename –zone-file-format example.zone" [2]

To modify record Transactions are used, a transaction is group of one or more record changes that should be propagated together, which ensures the data is never saved partially. To Modify DNS records use has to first start the transaction e.g. "gcloud dns record-sets transaction start -z=examplezonename" [2]

As the transaction start cloud DNS creates local file in YAML format as "transaction.yaml" each specified operation gets added to this file. e.g. gcloud dns record-sets transaction add -z=examplezonename –name="mail.example.com." –type=A –ttl=300 "7.5.7.8" [2]

3.4. Supported DNS record types

Table 1 shows an example table.

[2]

Table 1. Cloud DNS supports the following types of records:

Record type	Description
A	Address record, which is used to map host names to their IPv4 address.
AAAA	IPv6 Address record, which is used to map host names to their IPv6 address.
CAA	Certificate Authority (CA) Authorization, which is used to specify which CAs are allowed to create certificates for a domain.
CNAME	Canonical name record, which is used to specify alias names.
MX	Mail exchange record, which is used in routing requests to mail servers.
NAPTR	Naming authority pointer record, defined by RFC3403.
NS	Name server record, which delegates a DNS zone to an authoritative server.
PTR	Pointer record, which is often used for reverse DNS lookups.
SOA	Start of authority record, which specifies authoritative information about a DNS zone. An SOA resource record is created for you when you create your managed zone. You can modify the record as needed.
SPF	Sender Policy Framework record, a deprecated record type formerly used in e-mail validation systems (use a TXT record instead).
SRV	Service locator record, which is used by some voice over IP, instant messaging protocols, and other applications.
TXT	Text record, which can contain arbitrary text and can also be used to define machine-readable data, such as security or abuse prevention information.

4. SERVICES PROVIDED BY GOOGLE CLOUD DNS

Google Cloud DNS provides full control over DNS management and services, with Google's command and exposed REST APIs

user can manage their zones, records, migrate DNS from non-cloud to cloud DNS. Based on the SLA defined in the service plans user purchase the guaranty of services is provided. Apart from the basic plans user has 24X7 support from googles' expert team to resolved query or any issue user is facing. Apart from technical support team user can carry out maintenance of their own by referring to the technical documents and guidelines.

4.1. Migrating to Cloud DNS

Cloud DNS supports the migration of an existing DNS domain from another DNS provider to Cloud DNS just by creating Managed zones from user's domain, importing existing DNS configuration, verify DNS propagation, and updating registrar's name service records.

4.2. Monitoring Changes

DNS changes can be done via command line tool or REST API, they are initially marked as pending and user can verify the changes has reflected by referring to change history or look for status change. Listing changes and verifying DNS propagation by using the watch and dig commands to monitor changes has picked up by DNS server.

Syntax for lookup zone's server name: "gcloud dns managed-zones describe $< zonename >$"[3]

With this command it'll list down all the zone name server within that zone, which can be used to monitor individual server with Syntax. "watch dig example.com in MX @ $< yourzone's nameserver >$"[3]

The watch command will run the dig command every 2 seconds by default. Migration to Cloud DNS and Monitoring are activities the users can easily carry out by them selves

5. SCALABILITY

Google Cloud DNS inherently scaled for larger data set, as every solutions provided by Google Cloud are backed up by the googles' wide spread highly scalable infrastructure of Google compute Engine. Google's auto scaling and load balancing technique compute resources can be distributed over the multiple regions

"With Google's cloud load balancing user can put resources behind single anycast IP and scale resources up and down with intelligent autoscaling. It provides cross-region load balancing including automatic multi-region failover which gently moves traffic infractions if backend become unhealthy"[4]

6. GOOGLE CLOUD DNS WITH BIG DATA

With Google's one account user gets to use multiple cloud services that google is offering. Once the user creates the account and starts using the Google's compute engine and Appl engine, they are closer to harness the power of Big Data solutions offered on cloud platform, and with cloud DNS's programable feature it is up to user to leverage power of Big data analytics and enhance the caching of DNS resolver. Which is inherently implemented by Google themselves.

7. CLOUD DNS SERVICES IN COMPARISON WITH GOOGLE

In comparison with other cloud based DNS services offerings Google is pretty new in their comparison, and the cloud DNS market is already well established and customers get to choose appropriate service as their requirement. Few of the well-known cloud DNS provider with Key feature and services to offer to end customer.

Cloudflare Global DNS is free and fast in comparison and stand as no one position as per the market share. With its highly available DNS infrastructure and global anycast network address latency issue with local and global load balancer and health checks with fast failover to reroute visitors and website is always available.

Amazon Route 53 is amazon's cloud DNS service gives their customer option along with their other cloud services along with AWS. However this is comparatively expensive service as compared with other providers.

Microsoft Azure DNS, is said to be most cheapest cloud DNS however along with Google Cloud it is newly launch in DNS market and has about similar market share compared with google.

8. KEY POINTS TO TAKE AWAY

Google Cloud DNS is comparatively new in the market however with its competitive pricing and highly available and vast spread infrastructure it is challenging the market especially to Amazon's Route 53 services. With Google's technical Support, documentations, online help. User can easily migrate their DNS and choose cloud option. With Google's one account to it is easy to maintain multiple services under one roof. Google is eyeing to become the market leader in all Cloud Services as they has to offer with Cloud DNS resolver it ensures the every web site using their DNS services are always up and running 24X7 that wins customer's confidence. Google Cloud DNS though it is new in the market but with it's competitive pricing and services it is moving towards becoming the leader in services of Cloud DNS

REFERENCES

[1] "Google public dns," Web Page, 2017. [Online]. Available: https://developers.google.com/speed/public-dns/docs/intro
[2] "Cloud dns overview," Web Page, 2016. [Online]. Available: https://cloud.google.com/dns/overview
[3] "Cloud dns managing zones," Web Page, 2017. [Online]. Available: https://cloud.google.com/dns/zones/
[4] "Google cloud load balancing," 2017. [Online]. Available: https://cloud.google.com/load-balancing/

AUTHOR BIOGRAPHY

Vishwanath Kodre received his Masters Degree in Computer Science from Pune University. He is currently studying Data Science at Indiana University Bloomington.

Robot Operating System (ROS)

MATTHEW LAWSON[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: laszewski@gmail.com*

paper2, April 30, 2017

The Open Source Robotics Foundation (OSRF) oversees the maintenance and development of the Robot Operating System (ROS). ROS provides an open-source, extensible framework upon which roboticists can build simple or highly complex operating programs for robots. Features to highlight include: a) ROS' well-developed, standardized intra-robot communication system; b) its sufficiently-large set of programming tools; c) its C++ and Python APIs; and, d) its extensive library of third-party packages to address a large proportion of roboticists software needs. The OSRF distributes ROS under the BSD-3 license.

Keywords: Cloud, I524, robot, ros, ROS

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IO-3010/report.pdf

1. INTRODUCTION

The Open Source Robotics Foundation's middleware product *Robot Operating System*, or ROS, provides a framework for writing operating systems for robots. ROS offers "a collection of tools, libraries, and conventions [meant to] simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms" [2]. The Open Source Robotics Foundation, hereinafter OSRF or the Foundation, attempts to meet the aforementioned objective by implementing ROS as a modular system. That is, ROS offers a core set of features, such as inter-process communication, that work with or without pre-existing, self-contained components for other tasks.

2. ARCHITECTURE

The OSRF designed ROS as a distributed, modular system. The OSRF maintains a subset of essential features for ROS, i.e., the core functions upon which higher-level packages build, to provide an extensible platform for other roboticists. The Foundation also coordinates the maintenance and distribution of a vast array of ROS add-ons, referred to as modules. Figure 1 illustrates the ROS universe in three parts: a) the plumbing, ROS' communications infrastructure; b) the tools, such as ROS' visualization capabilities or its hardware drivers; and c) ROS' ecosystem, which represents ROS' core developers and maintainers, its contributors and its user base.

The modules or packages, which are analogous to packages in Linux repositories or libraries in other software distributions such as *R*, provide solutions for numerous robot-related challenges. General categories include a) drivers, such as sensor and actuator interfaces; b) platforms, for steering and image processing, etc.; c) algorithms, for task planning and obstacle avoidance; and, d) user interfaces, such as tele-operation and sensor data display. [3]

2.1. Communications Infrastructure

2.1.1. General

OSRF maintains three distinct communication methods for ROS: a) *message passing*; b) *services*; and, c) *actions*. Each method utilizes ROS' standard communication type, the *message* [4]. Messages, in turn, adhere to ROS' *interface description language*, or IDL. The IDL dictates that messages should be in the form of a data structure comprised of typed fields [5]. Finally, *.msg* files store the structure of messages published by various nodes so that ROS' internal systems can generate source code automatically.

2.1.2. Message Passing

ROS implements a publish-subscribe anonymous message passing system for inter-process communication, hereinafter pubsub, as its most-basic solution for roboticists. A pubsub system consists of two complementary pieces: a) a device, node or process, hereinafter node, publishing messages, i.e., information, to a *topic*; and b) another node *listening to* and ingesting the information from the associated topic. Designating topics to which a node should subscribe and topics to which a node should publish falls to the roboticist. ROS' *rosnode* command line tool conveniently "display[s] a list of active topics, the publishers and subscribers of a specific topic, the publishing rate of a topic, the bandwidth of a topic, and messages published to a topic" [6].

Pubsub's method of operation analogizes to terrestrial radio. In the analogy, the radio station represents the publishing node, the radio receiver maps to the subscribing node and the frequency on which one transmits and the other receives repre-

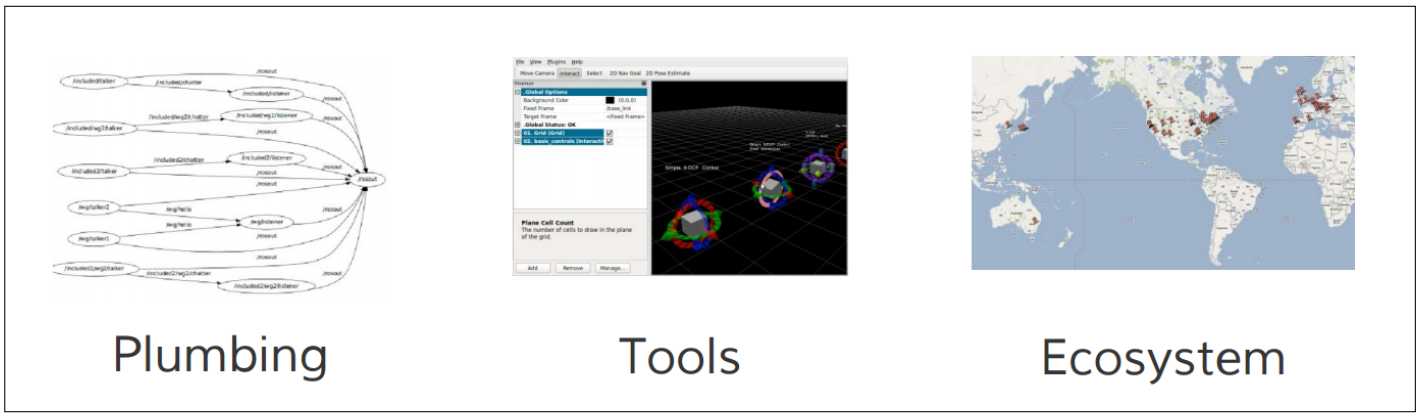


Fig. 1. A Conceptualization of What ROS, the *R*obot *O*perating *S*ystem, Offers to Roboticists [1]

sents the topic. Unlike terrestrial radio, though, ROS provides a lookup mechanism versus "flipping through the dial."

The OSRF touts the pubsub communications paradigm as the ideal method primarily due to its anonymity and its requirement to communicate using its message format. With respect to the first point, the nodes involved in bilateral or multilateral conversations need only know the topic on which to publish or subscribe in order to communicate. As a result, nodes can be replaced, substituted or upgraded without changing a single line of code or reconfiguring the software in any manner. The subscriber node can even be deleted entirely without affecting any aspect of the robot except those nodes that depend on the deleted node.

In addition, ROS' pubsub requires well-defined interfaces between nodes in order to succeed. For instance, if a node publishes a message without a crucial piece information a subscribing node requires or in an unexpected format, the message would be useless. Alternatively, it would be pointless for an audio processing node to subscribe to a node publishing lidar data. Therefore, a message's structure must be well-defined and available for reference as needed in order to ensure compatibility between publisher and subscriber nodes. As a result, ROS has a modular communication system. That is, a subscriber node may use all or only parts of a publishing node's message. Further, the subscribing node can combine the data with information from another node before publishing the combined information to a different topic altogether for a third node's use. At the same time, a fourth and fifth node could subscribe to the original topic for each node's respective purpose.

Finally, ROS' pubsub can natively replay messages by saving them as files. Since a subscriber node processes messages received irrespective of the message's source, publishing a saved message from a subscriber node at a later time works just as well as an actual topic feed. One use of asynchronous messaging: postmortem analysis and debugging.

2.1.3. Services

ROS also provides a synchronous, real-time communication tool under the moniker *services* [7]. Services allow a subscribing node to request information from a publishing node instead of passively receiving whatever the publishing node broadcasts whenever it broadcasts it. A service consists of two messages, the request and the reply. It otherwise mirrors ROS' message passing function. Finally, users can establish a continuous connection between nodes at the expense of service provider flexibility.

2.1.4. Actions

ROS *actions* offer a more-advanced communication paradigm than either message passing or services [8]. Actions, which use the basic message structure from message passing, allow roboticists to create a request to accomplish some task, receive progress reports about the task completion process, receive task completion notifications and / or cancel the task request. For example, the roboticist may create a task, or equivalently, initiate an action, for the robot to conduct a laser scan of the area. The request would include the scan parameters, such as minimum scan angle, maximum scan angle and scan speed. During the process, the node conducting the scan will regularly report back its progress, perhaps as a value representing the percent of the scan completed, before returning the results of the scan, which should be a point cloud. Roboticists can program a ROS-driven robot to attempt to accomplish any task imaginable, subject to physical realities. Sample actions: a) move X meters left; b) detect the handle of a door; and / or c) locate an empty soda can on a table, pick it up to determine if it is empty enough to recycle, crush it if it is empty enough, identify the recyle bin and then place it in the recylce bin [9].

2.2. Tools

2.2.1. Message Standards

ROS' extensive use in the robotics realm has allowed it to create message standards for various robot components [4]. In this case, *message standards* refers to expectations regarding information and information types robot components will provide to subscribing nodes. For instance, standard message definitions exist "for geometric concepts like poses, transforms, and vectors; for sensors like cameras, IMUs and lasers; and for navigation data like odometry, paths, and maps; among many others." These standards facilitate interoperability amongst robot components as well as easing development efforts by roboticists.

2.2.2. Robot Geometry Library

Robots with independently movable components, such as appendages (with joints) or movable sensors, must be able to coordinate such movements in order to be usable. Maintaining an accurate record of where a movable component is in relation to the rest of the robot presents a significant challenge in robotics [4].

ROS addresses this issue with its *transform* library. The *tf* library tracks components of a robot using three-dimensional coordinate frames [10]. It records the relationship between co-

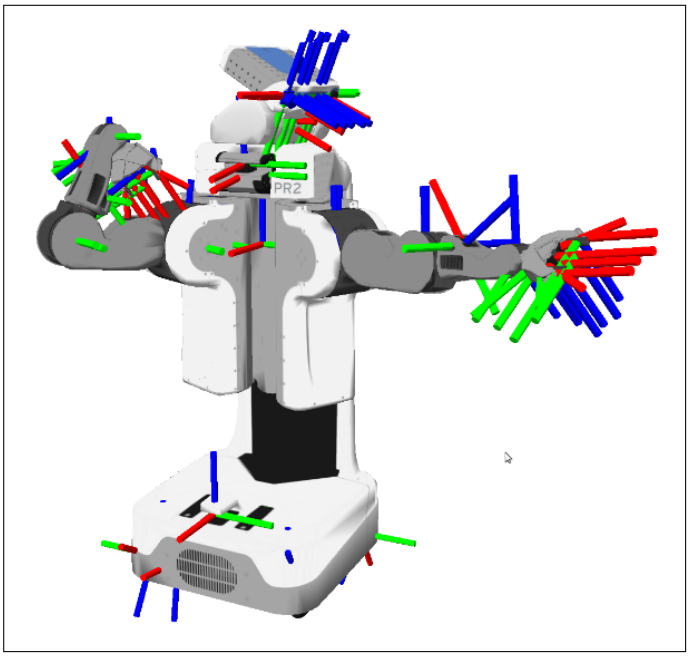


Fig. 2. A Simulated Robot with Many Coordinate Frames [10]

ordinate frame positional values at sequential points in time in a tree structure. tf's built-in functions allow the roboticist to transform a particular coordinate frame's values to same basis as a different coordinate frame's values. As a result, the user, or the user's program, can always calculate any coordinate frame's relative position to any or all of the other coordinate frame positions at any point in time. Although the first-generation library, *tf*, has been deprecated in favor of the second-generation one, *tf2*, the Foundation and ROS users still refer to the library as tf.

2.2.3. Robot Description Language

ROS describes robots in a machine-readable format using its *Unified Robot Description Format*, or URDF [4]. The file delineates the physical properties of the robot in XML format. URDF files enable use of the tf library, useful visualizations of the robot and the use of the robot in simulations.

2.2.4. Diagnostics

ROS' diagnostics meta-package, i.e., a package of related packages, "contains tools for collecting, publishing, analyzing and viewing diagnostics data [11]." ROS' diagnostics take advantage of the aforementioned message system to allow nodes to publish diagnostic information to the standard diagnostic topic. The nodes use the diagnostic_updater and self_test packages to publish diagnostic information, while users can access the information using the rqt_robot_monitor package. ROS does not require nodes to include certain information in their respective publications, but diagnostic publications generally provide some standard, basic information. That information may include serial numbers, software versions, unique incident IDs, etc.

2.2.5. Command Line Interfaces (CLI)

ROS provides at least 45 command line tools to the roboticist [12]. Therefore, ROS can be setup and run entirely from the command line. However, the GUI interfaces remain more popular among the user-base. Examples of ROS CLI tools include: a) rosmsg , which allows the user to examine messages, including the data structure of .msg files [5]; b) rosbag , a tool to perform various operations on .bag files, i.e., saved node publications; and, c) rosbash , which extends bash, a Linux shell program, with ROS-related commands.

2.2.6. Graphical User Interfaces (GUI)

OSRF includes two commonly-used GUIs, *rviz* and *rqt* [4], in the core ROS distribution. rviz creates 3D visualizations of the robot, as well as the sensors and sensor data specified by the user. This component renders the robot in 3D based on a user-supplied URDF document. If the end-user wants or needs a different GUI, s/he can use rqt, a Qt-based GUI development framework. It offers plug-ins for items such as: a) viewing layouts, like tabbed or split-screens; b) network graphing capabilities to visualize the robot's nodes; c) charting capabilities for numeric values; d) data logging displays; and, e) topic (communication) monitoring.

2.3. Ecosystem

ROS benefits from a wide-ranging network of interested parties, including core developers, package contributors, hobbyists, researchers and for-profit ventures. Although quantifiable use metrics for ROS remain scarce, ROS does have more than 3,000 software packages available from its community of users [13], ranging from proof-of-concept algorithms to industrial-quality software drivers. Corporate users include large organizations such as Bosch (Robert Bosch GmbH) and BMW AG, as well as smaller companies such as ClearPath Robotics, Inc. and Stanley Innovation. University users include the Georgia Institute of Technology and the University of Arizona, among others [14].

3. API

ROS supports robust application program interfaces, APIs, through libraries for C++ and Python. It provides more-limited, and experimental, support for nodejs, Haskell and Mono / .NET programming languages, among others. The latter library opens up use with C# and Iron Python [15].

4. LICENSING

The OSRF distributes the core of ROS under the standard, three-clause BSD license, hereinafter BSD-3 license. The BSD-3 license belongs to a broader class of copyright licenses referred to as *permissive licenses* because it imposes zero restrictions on the software's redistribution as long as the redistribution maintains the license's copyright notices and warranty disclaimers [16].

Other names for BSD-3 include: a) BSD-new; b) New BSD; c) revised BSD; d) The BSD License, the official name used by the Open Source Initiative; and, e) Modified BSD License, used by the Free Software Foundation.

Although the OSRF distributes the main ROS elements under the BSD-3 license, it does not require package contributors or end-users to adopt the same license. As a result, full-fledged ROS programs may include other types of *Free and Open-Source Software* [17], or FOSS, licenses. In addition, programs may depend on proprietary or unpublished drivers unavailable to the broader community.

5. USE CASES

ROS' end-markets, its use cases, include manipulator robots, i.e., robotic arms with grasping units; mobile robots, such as autonomous, mobile platforms; autonomous cars; social robots; humanoid robots, unmanned / autonomous vehicles; and an assortment of other robots [14].

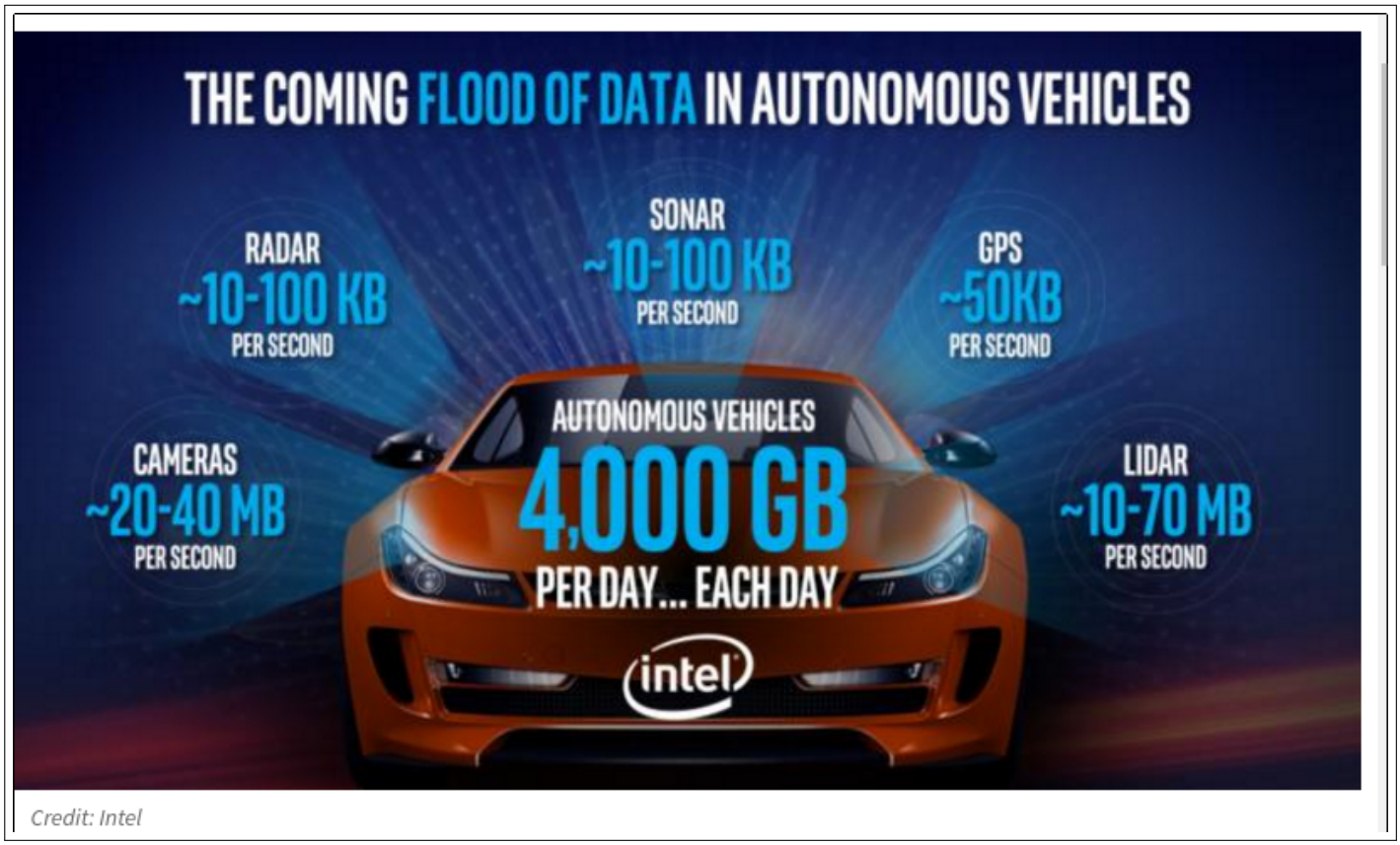


Fig. 3. Estimated Amount of Data Produced by an Autonomous Automobile [18]

5.1. Automated Data Collection Platform

Fetch Robotics, Inc. offers its *Automated Data Collection Platform* robot to the market so corporations can "[g]ather environmental data more frequently and more accurately for [its] Internet of Things...and Big Data Applications [19]." Fetch's system automatically collects data, such as RFID tracking (inventory management) or in-store shelf surveys. The latter service began in January 2017 when Fetch partnered with Trax Image Recognition, which makes image recognition software [20].

5.2. Autonomous Robots of Various Natures

The applications of autonomous robots abound, especially for monotonous or dangerous work. For instance, ClearPath Robotics, a company that sells autonomous robots using ROS, currently highlights the following use cases: a) "geotechnical mapping of rock masses, a crucial step in predicting potentially lethal rock falls and rock bursts in and around mines [21];" minesweeping [22]; and vibration control of bridges [23]. In order to accomplish any of the aforementioned tasks, or any of the other tasks, the robots need sensors, such as lidar, high-definition cameras and / or sonar. One estimate places lidar and camera data production at 10MB to 70MB per second and 20MB to 40MB per second [18]. Given the multitude of addressable end markets, as well as the probability of multiple robots with multiple sensors associated with one effort in any particular end market, the data produced by a ROS robot vaults it into *big data* territory. A single ROS instance may never create 4,000GB of data per day like an autonomous consumer automobile, but a small swarm of moderately complex robots run by a three-person team will likely prompt the team to avail itself of big data techniques.

6. CONCLUSION

ROS offers a number of attractive features to its users, including a well-developed and standardized intra-robot communication system, modular design, vetted third-party additions and legitimacy via real-world applications. Although its status as open source software precludes direct support from its parent organization, OSRF, for-profit organizations and the software's active community of users provide reassurances to any roboticist worried about encountering a seemingly-insurmountable problem.

7. ACKNOWLEDGMENT

I would like to thank my employer, Indiana Farm Bureau, for its support of my continuing education.

REFERENCES

[1] H. Boyer, "Open Source Robotics Foundation And The Robotics Fast Track," web page, nov 2015, accessed 19-mar-2017. [Online]. Available: https://www.osrfoundation.org/wordpress2/wp-content/uploads/2015/11/rft-boyer.pdf

[2] Open Source Robotics Foundation, "About ROS," Web page, mar 2017, accessed 16-mar-2017. [Online]. Available: http://www.ros.org/about-ros/

[3] National Instruments, "A Layered Approach to Designing Robot Software," Web page, mar 2017, accessed 18-mar-2017. [Online]. Available: http://www.ni.com/white-paper/13929/en/

[4] Open Source Robotics Foundation, "Why ROS?: Features: Core components," Web page, mar 2017, accessed 17-mar-2017. [Online]. Available: http://www.ros.org/core-components/
[5] Open Source Robotics Foundation, "ROS graph concepts: Messages," Web page, aug 2016, accessed 18-mar-2017. [Online]. Available: http://wiki.ros.org/Messages
[6] Open Source Robotics Foundation, "rostopic: Package Summary," Web page, jun 2016, accessed 09-apr-2017. [Online]. Available: http://wiki.ros.org/rostopic
[7] Open Source Robotics Foundation, "Services," web page, feb 2012, accessed 18-mar-2017. [Online]. Available: http://wiki.ros.org/Services
[8] Open Source Robotics Foundation, "actionlib: Package summary," Web page, feb 2017, accessed 18-mar-2017. [Online]. Available: http://wiki.ros.org/actionlib
[9] MathWorks, Inc. The, "Control PR2 Arm Movements Using ROS Actions and Inverse Kinematics," Web page, apr 2017, accessed 09-apr-2017. [Online]. Available: https://goo.gl/3tZqow
[10] Open Source Robotics Foundation, "tf2: Package summary," Web page, mar 2016, accessed 18-mar-2017. [Online]. Available: http://wiki.ros.org/tf2
[11] Open Source Robotics Foundation, "diagnostics: Package Summary," Web page, aug 2015, accessed 19-mar-2017. [Online]. Available: http://wiki.ros.org/diagnostics
[12] Open Source Robotics Foundation, "ROS command-line Tools," Web page, aug 2015, accessed 19-mar-2017. [Online]. Available: http://wiki.ros.org/ROS/CommandLineTools
[13] Open Source Robotics Foundation, "Why ROS?: Is ROS for me?" Web page, mar 2017, accessed 16-mar-2017. [Online]. Available: www.ros.org/is-ros-for-me/
[14] Open Source Robotics Foundation, "Robots Using ROS," Web page, mar 2017, accessed 19-mar-2017. [Online]. Available: http://wiki.ros.org/Robots
[15] Open Source Research Foundation, "APIs," Web page, mar 2016, accessed 19-mar-2017. [Online]. Available: http://wiki.ros.org/APIs
[16] Wikipedia, "BSD licenses — Wikpedia, the free encyclopedia," Web page, mar 2017, accessed 16-mar-2017. [Online]. Available: https://en.wikipedia.org/wiki/BSD_licenses
[17] Wikipedia, "Free and open-source software – Wikipedia, the free encyclopedia," Web page, mar 2017, accessed 18-mar-2017. [Online]. Available: https://en.wikipedia.org/wiki/Free_and_open-source_software
[18] P. Nelson, "Just one autonomous car will use 4,000 GB of data/day," Web page, dec 2016, accessed 09-apr-2017. [Online]. Available: https://goo.gl/HBL9GM
[19] Fetch Robotics, Inc., "Automated Data Collection Platform," Web page, mar 2017, accessed 19-mar-2017. [Online]. Available: http://fetchrobotics.com/automated-data-collection-platform/
[20] Retail Touch Points, "Trax and Fetch Robotics Unite for Better Retail," Web page, jan 2017, accessed 19-jan-2017. [Online]. Available: http://www.retailtouchpoints.com/features/news-briefs/trax-and-fetch-robotics-unite-for-better-retail-insights
[21] Clearpath Robotics Inc., "Husky performs slam on stereonets to help predict rock falls and rock bursts," Web page, apr 2017, accessed 09-apr-2017. [Online]. Available: https://www.clearpathrobotics.com/husky-queens-mining-slam-stereonets/
[22] Clearpath Robotics Inc., "Bomb-detecting husky to remove killer landmines," Web page, apr 2017, accessed 09-apr-2017. [Online]. Available: https://www.clearpathrobotics.com/coimbra-autonomous-demining-husky/
[23] Clearpath Robotics Inc., "Deployable, autonomous vibration control of bridges using husky ugv," Web page, apr 2017, accessed 09-apr-2017. [Online]. Available: https://www.clearpathrobotics.com/autonomous-vibration-control-bridges-using-husky-ugv/

AUTHOR BIOGRAPHIES

Matthew Lawson received his BSBA, Finance in 1999 from the University of Tennessee, Knoxville. His research interests include data analysis, visualization and behavioral finance.

A. WORK BREAKDOWN

The work on this project was distributed as follows between the authors:

Matthew Lawson. Matthew researched and wrote all of the material for this paper.

Apache Crunch

SCOTT MCCLARY[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: scmcclar@indiana.edu

paper-002, April 30, 2017

Apache Crunch is an Application Programming Interface (i.e. API) designed for the Java programming language. This software is built on top of Apache Hadoop as well as Apache Spark and simplifies the process of developing MapReduce pipelines. Apache Crunch abstracts away the explicit need to manage MapReduce jobs. This defining characteristic alleviates much of the steep learning curve inherently within developing scalable applications that utilize a MapReduce type approach. Therefore, developers using Apache Crunch are able to streamline the process of converting Big Data solutions into runnable code. As a result, this Java API is leveraged in industry and academia to develop efficient, scalable and maintainable codebases for Big Data solutions.

Keywords: Big-Data, Cloud, Hadoop, MapReduce

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IO-3011/report.pdf

1. INTRODUCTION

Apache Crunch is an open source Java API that is "built for pipelining MapReduce programs which are simple and efficient;" more specifically, Crunch allows developers to write, test and run MapReduce pipelines with minimal upfront investment [1, 2]. The minimal upfront investment of this API lowers the barrier for entry for Big Data developers.

Apache Crunch's purpose is to make "writing, testing, and running MapReduce pipelines easy, efficient, and even fun" [2, 3]. This open source Java API provides a "small set of simple primitive operations and lightweight user-defined functions that can be combined to create complex, multi-stage pipelines" [3]. Apache Crunch abstracts away much of the complexity from the user by compiling "the pipeline into a sequence of MapReduce jobs and manages their execution" [2, 3].

1.1. History

Josh Wills at Cloudera was the major contributor/developer to the Crunch project in 2011 [4, 5]. The original version of this software was based on Google's FlumeJava library [4–6]. From 2011 until May 2012 (i.e. version 0.2.4), the Apache Crunch project was open sourced at GitHub [4, 7]. After May 2012, the original Crunch source code was donated to Apache by Cloudera and shortly after "the Apache Board of Directors established the Apache Crunch project in February 2013 as a new top level project" [4]. Since February 2013, the Apache Crunch project continues to be used, maintained and improved in an open source fashion by the software's user and developer community. The user community has grown to include large reputable companies such as Spotify and Cerner [8, 9].

1.2. Advantages

As Hadoop continues to grow in popularity, the variation of data (i.e. satellite images, time series data, audio files, and seismograms) that is stored in HDFS grows as well [3, 10]. Many of these data "formats are not a natural fit for the data schemas imposed by Pig and Hive;" therefore, "large, custom libraries of user-defined functions in Pig or Hive" or "MapReduces in Java" have to be written, which significantly "drain on developer productivity" [2, 3, 11, 12]. The Crunch API provides an alternative solution, which does not inhibit developer productivity. Apache crunch integrates seamlessly into Java and therefore, allows developers full access to Java to write functions. Thus, Apache Crunch is "especially useful when processing data that does not fit naturally into relational model, such as time series, serialized object formats like protocol buffers or Avro records, and HBase rows and columns" [13–15].

1.3. API

Apache Crunch is a Java API that is used "for tasks like joining and data aggregation that are tedious to implement on plain MapReduce" [13]. The Apache Software Foundation provides thorough documentation of the API for Apache Crunch and even provides useful examples of how to explicitly leverage this API from a Java application [13].

1.3.1. Shell Access

For users of the Scala programming language, there is the "Scrunch API, which is built on top of the Java APIs and in-

cludes a REPL (read-eval-print loop) for creating MapReduce pipelines" [13].

2. LICENSING

The Apache Software Foundation, which includes the software tool named Apache Crunch, is licensed under the Apache License, Version 2.0 [16].

2.1. Source Code

Apache Crunch leverages Git for version control, which allows the user and developer communities to contribute freely to this open source project [7, 17].

3. ARCHITECTURE & ECOSYSTEM

In the simplest of terms, Apache Crunch runs on top of Hadoop MapReduce and Apache Spark [13]. Therefore, Apache Crunch abstracts away the need for the programmer to explicitly manage the MapReduce jobs through a Java API. However, the Apache Crunch's place within the software stack (i.e. on top of Hadoop MapReduce and Apache Spark) indicates its reliance on the MapReduce software subsystem. Given Apache Crunch's dependence on Hadoop MapReduce and Apache Spark, this API provides the ability for developers use the Java programming language to efficiently and effectively leverage MapReduce style processing to solve their complicated and complex Big Data problems.

4. USE CASES

Apache Crunch has its applicability in the Cloud Computing and Big Data industry, as shown in the following section. The widespread usage of Java, Apache Hadoop and Apache Spark in Cloud Computing help promote Apache Crunch in industry and academia alike.

4.1. Use Cases for Big Data

The Apache Hadoop ecosystem indicates that Apache Crunch is built on top of Hadoop MapReduce and Apache Spark, which both go hand in hand in solving many complicated and challenging Big Data problems. The following sections demonstrate Apache Crunch's applicability in Big Data problems. Furthermore, these use cases explains that the software facilitates the rapid and clean development of the respective Big Data solutions. The benefits realized at companies such as Cerner and Spotify are due in part to Apache Crunch's well-defined applicability in the Big Data space.

4.2. Cerner

Cerner, "an American supplier of health information technology (HIT) solutions, services, devices and hardware" [18], employs Apache Crunch to solve many of their Big Data problems [9]. Cerner decided to use Apache Crunch since it interestingly solves what they refer to as "a people problem" [9]. As a company, they have noticed that Apache Crunch diminishes a potential steep learning curve for new employees and/or teams to leverage Big Data technologies in their projects.

Cerner definitively believes that Apache Crunch stands above the other "options available for processing pipelines including Hive, Pig, and Cascading" since the Apache Crunch API allows their employees to straightforwardly code solutions to Big Data problems [9, 11, 12, 19]. The diminished learning curve as a result of using Apache Crunch allows Cerner to focus their time, effort and money on performance tuning and/or algorithm adjustments rather than wasting a significant amount the developers time simply translating a Big Data problem into runnable and efficient MapReduce code [9].

4.3. Spotify

Spotify, the popular "music, podcast, and video streaming service" [20], leverages Apache Crunch to process the many terabytes of data generated every day by their large user community [8]. Spotify has been using Apache Hadoop since 2009 and have spent significant effort since then to develop tools that make it simple for the Spotify "developers and analysts to write data processing jobs using the MapReduce approach in Python" [2, 8].

However, in 2013 Spotify came to the realization that this approach wasn't performing well enough so they decided to start using Java and Apache Crunch to solve their Big Data problems [8]. This transition to Apache Crunch resulted in higher performance, higher-level abstractions (e.g. filters, joins and aggregations), pluggable execution engines (e.g. MapReduce and Apache Spark) and added simple powerful testing (e.g. fast in-memory unit tests) [8]. Apache Crunch has given Spotify a significant enhancement for both their "developer productivity and execution performance on Hadoop" [8].

5. EDUCATIONAL MATERIAL

Apache Crunch makes the process of developing applications that leverage MapReduce and Apache Spark easier; therefore, the learning curve is much less significant in relation to developing applications that directly interact with MapReduce and Apache Spark. The Apache Software Foundation provides a lot of useful documentation. For instance, there is API documentation [21] as well as getting started information [22], a user guide [23] and even source code installation information [17]. If this is not enough, complete and extensive third-party code examples explain how to develop "hello world" applications that use Apache Crunch [24].

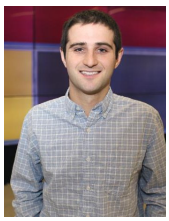
6. CONCLUSION

In general, Apache Crunch simplifies the process of writing and maintaining large-scale parallel codes by abstracting away the need to manage MapReduce jobs. This abstraction diminishes the inherent learning curve in solving Big Data problems and therefore allows developers to focus their time and effort in developing the general concept of their solution rather than in the detailed process of writing their code. The aforementioned benefits of Apache Crunch are proven by its widespread use in industry (e.g. Spotify and Cerner) and in academia. This software tool helps diminish the gap between domain scientists solving Big Data problems and the potentially complicated Computer Science tools/mechanisms provided to the Cloud Computing/Big Data community.

ACKNOWLEDGEMENTS

The authors would like to thank the School of Informatics and Computing for providing the Big Data Software and Projects (INFO-I524) course [25]. This paper would not have been possible without the technical support & edification from Gregor von Laszewski and his distinguished colleagues.

AUTHOR BIOGRAPHIES



Scott McClary received his BSc (Computer Science) and Minor (Mathematics) in May 2016 from Indiana University and will receive his MSc (Computer Science) in May 2017 from Indiana University. His research interests are within scientific application performance analysis on large-scale HPC systems. He will begin working as a Software Engineer with General Electric Digital in San Ramon, CA in July 2017.

WORK BREAKDOWN

The work on this project was distributed as follows between the authors:

Scott McClary. He completed all of the work for this paper including researching and testing Apache Airavata as well as composing this technology paper.

REFERENCES

[1] Edupristine, "Hadoop ecosystem and its components," Web Page, apr 2015, accessed: 2017-3-26. [Online]. Available: http://www.edupristine.com/blog/hadoop-ecosystem-and-components

[2] I. Wikimedia Foundation, "MapReduce - Wikipedia," Web Page, apr 2017, accessed: 2017-4-9. [Online]. Available: https://en.wikipedia.org/wiki/MapReduce

[3] J. Wills, "Introducing crunch: Easy mapreduce pipelines for apache hadoop," Blog, oct 2011, accessed: 2017-3-26. [Online]. Available: http://blog.cloudera.com/blog/2011/10/introducing-crunch/

[4] The Apache Software Foundation, "Apache Crunch - About," Web Page, 2013, accessed: 2017-3-26. [Online]. Available: https://crunch.apache.org/about.html

[5] Cloudera, Inc., "Big data | Machine Learning | Analytics | Cloudera," Web Page, 2017, accessed: 2017-4-09. [Online]. Available: https://www.cloudera.com

[6] C. Chambers, A. Raniwala, F. Perry, S. Adams, R. R. Robert R. Henry, R. Bradshaw, and N. Weizenbaum, "FlumeJava: Easy, Efficient Data-Parallel Pipelines," in *2010 ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI '10. Toronto, Ontario, Canada: ACM, 2010, pp. 363–375. [Online]. Available: http://doi.acm.org/10.1145/2609441.2609638

[7] GitHub, Inc., "GitHub," Web Page, 2017, accessed: 2017-4-09. [Online]. Available: https://github.com

[8] J. Kestelyn, "Data processing with apache crunch at spotify," Blog, feb 2015, accessed: 2017-3-26. [Online]. Available: http://blog.cloudera.com/blog/2015/02/data-processing-with-apache-crunch-at-spotify/

[9] M. Whitacre, "Scaling people with apache crunch," Blog, may 2014, accessed: 2017-3-26. [Online]. Available: http://engineering.cerner.com/blog/scaling-people-with-apache-crunch/

[10] The Apache Software Foundation, "Welcome to Apache Hadoop!" Web Page, mar 2017, accessed: 2017-4-9. [Online]. Available: http://hadoop.apache.org

[11] The Apache Software Foundation, "Welcome to Apache Pig!" Web Page, jun 2016, accessed: 2017-4-9. [Online]. Available: https://pig.apache.org

[12] The Apache Software Foundation, "Apache Hive TM," Web Page, 2014, accessed: 2017-4-9. [Online]. Available: https://hive.apache.org

[13] The Apache Software Foundation, "Apache Crunch Simple and Efficient MapReduce Pipelines," Web Page, 2013, accessed: 2017-3-26. [Online]. Available: https://crunch.apache.org

[14] The Apache Software Foundation, "Welcome to Apache Avro!" Web Page, may 2016, accessed: 2017-4-9. [Online]. Available: https://avro.apache.org

[15] The Apache Software Foundation, "Apache HBase – Apache HBase Home," Web Page, apr 2017, accessed: 2017-4-9. [Online]. Available: https://hbase.apache.org

[16] The Apache Software Foundation, "Apache license, version 2.0," Web Page, jan 2004, accessed: 2017-3-26. [Online]. Available: http://apache.org/licenses/LICENSE-2.0.html

[17] The Apache Software Foundation, "Getting the source code," Web Page, 2013, accessed: 2017-3-26. [Online]. Available: https://crunch.apache.org/source-repository.html

[18] I. Wikimedia Foundation, "Cerner - Wikipedia," Web Page, mar 2017, accessed: 2017-3-26. [Online]. Available: https://en.wikipedia.org/wiki/Cerner

[19] Xplenty Ltd, "Cascading | Application Platform for Enterprise Big Data," Web Page, 2016, accessed: 2017-4-9. [Online]. Available: http://www.cascading.org

[20] I. Wikimedia Foundation, "Spotify - Wikipedia," Web Page, mar 2017, accessed: 2017-3-26. [Online]. Available: https://en.wikipedia.org/wiki/Spotify

[21] The Apache Software Foundation, "Apache crunch 0.15.0 api," Web Page, 2017, accessed: 2017-3-26. [Online]. Available: https://crunch.apache.org/apidocs/0.15.0/

[22] The Apache Software Foundation, "Apache Crunch - Getting Started," Web Page, 2013, accessed: 2017-3-26. [Online]. Available: https://crunch.apache.org/getting-started.html

[23] The Apache Software Foundation, "Apache Crunch - Apache Crunch User Guide," Web Page, 2013, accessed: 2017-3-26. [Online]. Available: https://crunch.apache.org/user-guide.html

[24] N. Asokan, "Learn Apache Crunch," Blog, Mar 2015, accessed: 2017-3-26. [Online]. Available: http://crunch-tutor.blogspot.com

[25] Gregor von Laszewski and Badi Abdul-Wahid, "Big Data Classes," Web Page, Indiana University, Jan. 2017. [Online]. Available: https://cloudmesh.github.io/classes/

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, April 25, 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 more about the project are presented.

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 Leaonidas 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].

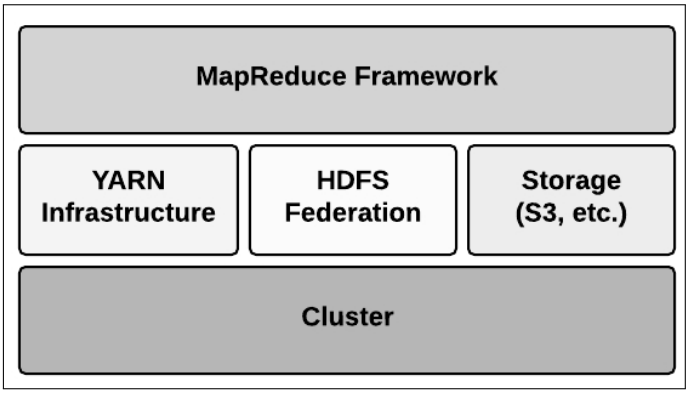


Fig. 1. Hadoop Components [5]

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].

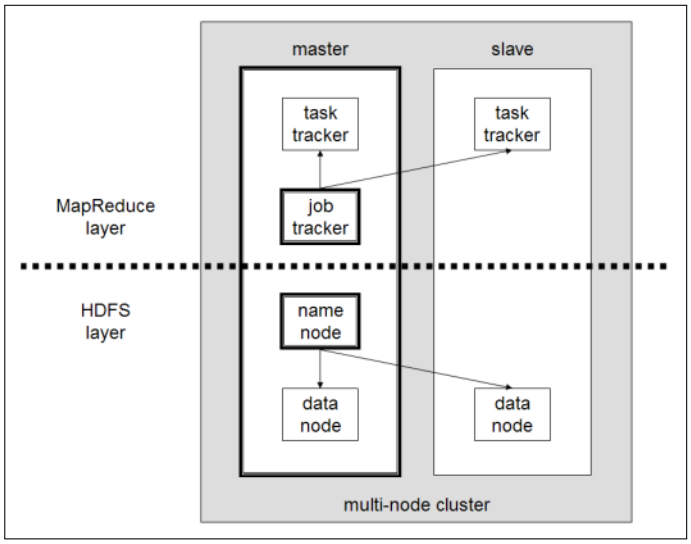


Fig. 2. Hadoop Cluster [4]

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 BCPMaster. Zookeeper efficiently manages synchronization of BSPPeers, instances started by groom servers [6].

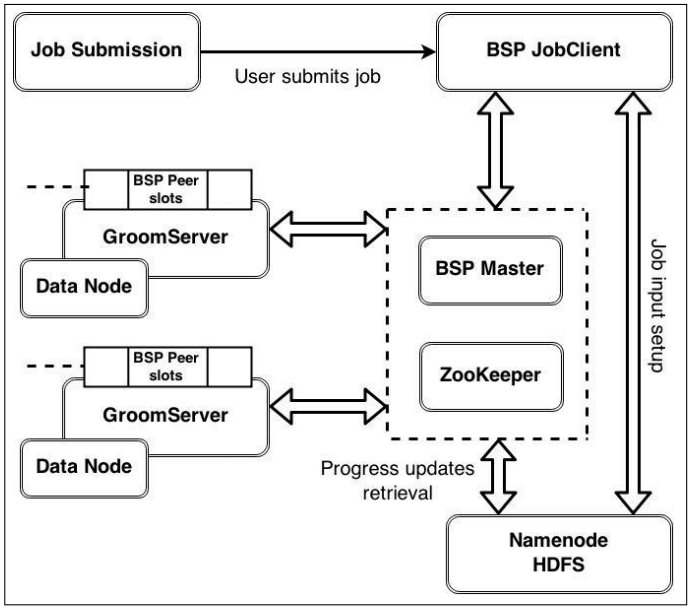


Fig. 3. Hama Architecture [7]

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].

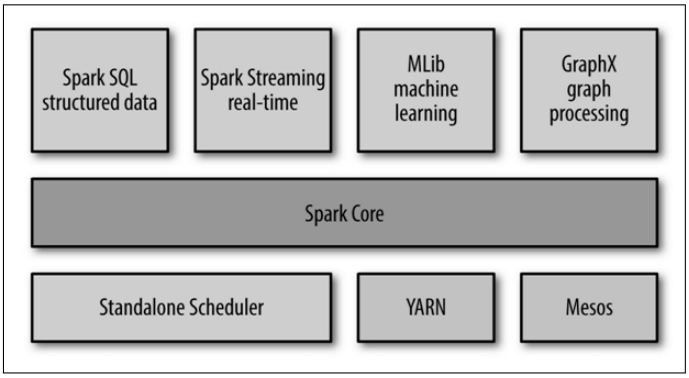


Fig. 4. Spark Architecture [9]

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 architechure is shown in Figure 5.

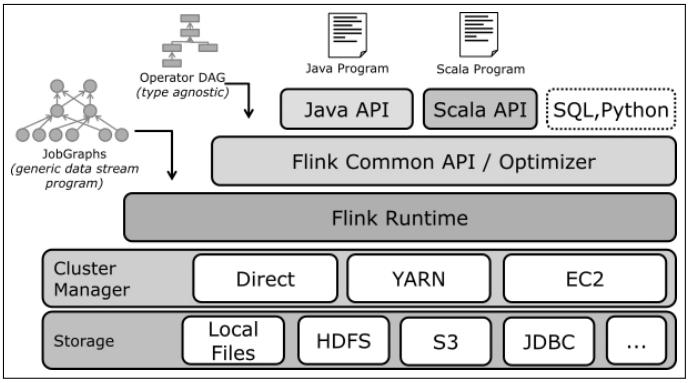


Fig. 5. Flink Architecture [11]

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 functionalily [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 btye code in the final jar file [12].

MAPREDUCE

The MapReduce programing 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 el ] ]
```

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].

Comparing MRQL to Hive, we find several differences. Hive stores metadata in a Relational Database Management System while MRQL does not utilize metadata. MRQL allows the use of Group By on arbitrary queries. Hive does not allow the use of Group By on subqueries. MRQL runs on Hadoop, Hama, Spark, and Flink while Hive works on Hadoop, Tez, and Spark. MRQL is compatible with text, sequence, XML, and JSON file formats. Hive is compatible with test, sequence, ORC, and RCFile formats. MRQL allows iteration; Hive does not. MRQL allows streaming; Hive does not [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 it freely available. It is stored in github [25].

CONCLUSION

MapReduce Query Language simplifies the popular MapReduce programing 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. Pandy, "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 Opearting 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 comples, 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)

[20] L. Fegaras, "Publications," Web Page, Apr. 2015, accessed 2017-03-24. [Online]. Available: https://wiki.apache.org/mrql/Publications

[21] L. Fegaras, "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] L. Fegaras, "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.

Lighting Memory-Mapped Database (LMDB)

LEONARD MWANGI[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: lmwangi@iu.edu

paper 2 April 30, 2017

LMDB is one of the most recent in-memory key-value databases, it has shown performance not matched by many other key-value databases and a unique capability to manage and reclaim unused space. We'll examine the database, it's architecture, features and use cases that makes it ideal for self-contained application in need a local small-footprint database.

Keywords: Cloud, I524, LMDB, BDB, B+tree

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IO-3013/report.pdf

1. INTRODUCTION

Lightning Memory-Mapped Database (LMDB) is a high-performance transaction non-relational database that stores data in the form of key-value store [1] while using memory-mapped file capabilities to increase I/O performance. Designed to solve multiple layers of configuration and caching issues inherent on Berkeley DB (DBD) design [2], LMDB fixes caching problems with a single, automatically managed cache which is controlled by the host operating system [3]. The database footprint is small enough to fit in an L1 cache [4] and its designed to always append data at the end of the database (MDB_APPEND) thus there is never a need to overwrite data or do garbage collection. This design protects the database from corruption in case of a system crash or need to carry out intensive recovery process.

LMDB is built on combination of multiple technologies dating back to 1960s. Memory-Mapped files or persistent object concept was first introduced by Multics in mid 1960s as single-level storage (SLS) [5] which provided distinction between files and process memory this technology incorporated in LMDB architecture. LMDB also utilizes B+tree architecture which provides a self-balancing tree data structure making it easy to search, insert and delete data due to its sorting algorithm. LMDB specifically utilizes the append-only B+tree code written by Martin Hedenfalk [6] for OpenBSD ldapd implementation. The architecture is also modeled after BDB API and perceived as it's replacement.

2. ARCHITECTURE

The following section provides an overview of LMDB architecture and the technologies involved.

2.1. Language

LMDB is written in C but the API supports multiple programming languages with C and C++ supported directly while other languages like Python, Ruby, Erlang amongst others supported using wrappers [7].

2.2. B+tree

The append-only B+tree architecture ensures that that the data is never overwritten thus every time modification is required, a copy is made and changes made to the copy which is in turn written to a new disk space. To reclaim the freed space, LMDB uses a second B+tree which keeps track of pages that have been freed thus keeping the database size fairly the same. This is a major architectural advantage compared to other B+tree based databases.

Figure 2 Regular B+tree architecture.

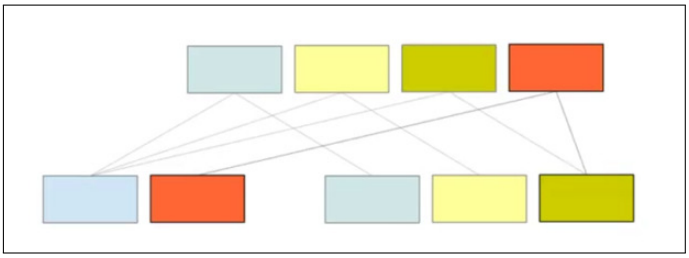


Fig. 1. For every new leaf-node created, a root-node is created

Figure 2 LMDB B+tree architecture.

2.3. Transaction

The database is architected to support single-writer and many-readers per transaction [8] ensuring no deadlock within the database. No memory allocation (malloc) or memory-to-memory copies (memcpy) required by the database thus ensuring that locks do not occur during

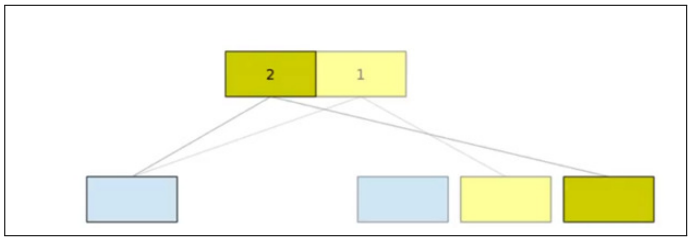


Fig. 2. LMDB uses two root-nodes for valid leaf-nodes as shown in diagram above. Unused leaf-nodes do not have root-nodes tied to them which signifies unused page that can be reclaimed

3. IMPLEMENTATION

LMDB implementation is supported by both Windows and Unix operating systems. It is provided in two variants which are automatically selected during installation. CFFI variant supports PyPy for CPython version 2.6 or greater and C extensions that support CPython versions 2.5 through 2.7 and version 3.3 and greater. For ease of adoption, both versions have the same interface. This installation guide will focus on Python based implementation for both operating systems. At the moment 32-bit and 64-bit binaries are provided for Python 2.7. Future binaries are expected to support all versions of Python.

3.1. Installation

Installation and configuration process for LMDB.

```
#Unix based Systems
pip install lmdb
#or
easyinstall lmdb
#Create and open and environment
 mdb_env_create() mdb_env_open()
 #defines the directory path and must
 exist before being used.
 #mdb_nosubdir option can be used if no
 directory path needs to be passed.
#Create transactions after opening the
# environment
 mdb_env_create() mdb_env_open()
 #defines the directory path and must
 exist before being used.
 #mdb_nosubdir option can be used if no
 directory path needs to be passed.
#Open database for the transaction
#mdb_dbi_open() #NULL value can be passed
    if only a single database
    will be used in the environment.
#mds_create flag must be specified to
     create a named database.
#mdb_env_set_maxdbs() defines the maximum
    number of databases the
    environment will support.
#mdb_get() and #mdb_put() can be
    used to store a single key/value pairs,
    if more transactions are needed ver
    then cursors are required.
#mdb_txn_commit() is required to
    commit the data in the environment
```

4. FEATURES

LMDB has the following features that are not in a regular key/value store databases:

1. Explicit Key Types Key comparisons in reverse byte order as well as native binary integer supported, this goes beyond the regular key/value string comparisons and memory-to-memory copy.

2. Append Mode Useful when bulk loading the data which is already a sorted format otherwise it would lead to data corruption.

3. Fixed Mapping Data can be stored in a fixed memory address where applications connecting to the database will see all the objects with the same address.

4. Hot Backups Since LMDB uses MVCC, a hot backup can be carried while the database is live because transactions are self-consisted from beginning to the end.

5. LICENSING

LMDB is licensed under OpenLDAP public license, a BSD style licensing.

6. ALTERNATIVES

There are several alternatives to LMDB, focusing on key-value databases, some of the leading alternatives include the following:

6.1. SQLite3

One of the alternatives to LMDB, an in-process database designed to be embedded in the applications rather than having its own server. The main comparison with LMDB is they are both in-memory databases but SQLite3 is also a relational database with structured data in form of tables where is LMDB is not a relational database [9].

6.2. Oracle Berkeley DB (BDB)

A key-value database provided in form of software libraries offers an alternative to LMDB. LMDB is actually molded from DBD thus structure and architecture is almost the same. BDB is considered to be fast, flexible, reliable and scalable database and has the ability to access both key and sequential data. BDB does not require a stand-alone server and is directly integrated to the application [2].

6.3. Google LevelDB

LevelDB is an on-disk key-value open source database developed by google fellows and inspired by BigTable memtable/sstable architecture. One of the main differences from LMDB is LevelDB utilizes disk for storage rather than memory which can have negative impact on performance due to disk seeks [10]. Also, being an on-disk database, it is susceptible to corruption especially when there is system crash. LevelDB is licensed under BSD Licensing model [11].

6.4. Kyoto Cabinet

Provided in form of libraries, is an on-disk key-value database with B+tree and hash tables architecture. Kyoto Cabinet is written in C++ and has APIs for other languages [12]. Disk management has been identified as an issue for Kyoto Cabinet [13]. It's also licensed under General Public Licensing.

7. USE CASE

7.1. NoSQL Data Stores

Due to its small footprint and great performance, LMDB is an ideal candidate for NoSQL data stores. The data store is widely used for caching, queue-ing, tasks distribution and pub/sub to enhance performance. Being an in-memory database, LMDB is a great candidate for these stores and is currently being used as Redis data store [13].

7.2. Mobile Phone Database

In the wake of smart phones, the need for mobile applications to collect and store data has grown exponentially. Most of the mobile applications store this data in cloud requiring them to connect back and forth in order to facility user's request. Having a local lightweight self-contained database on the device has become an attractive solution to enhance user experience with performance and effectiveness. LMDB fills in this gap due to its small footprint, performance and ability to reuse storage thus making it a favorable mobile phone database.

7.3. Postfix Mail Transfer Agent (MTA) Database

Postfix is an SMTP Server that provides first layer of spambots and malware defense to the end users. Having the ability to track and keep history of the scans and identified threats at a fast rate is paramount for the success of Postfix. Postfix uses LMDB adapter to provide access to lookup tables and make data available to the application reliably [14].

8. CONCLUSION

Self-managing, small footprint and well performing database is a critical to many client side applications. These applications are better suited when they can offload storage management to the database application and gain great performance. LMDB has shown these capabilities by managing how data is written and claiming unused storage. It's great performance and small footprint stages it well for many of the application. In comparison to other databases in its class, LMDB benchmarks [15] shows its capability to be the leading database on in self-contained application databases. Also, being available as an OpenLDAP public licensing and having wrappers for different languages makes it easier to port to already existing applications.

ACKNOWLEDGEMENTS

This research was done as part of course "I524: Big Data and Open Source Software Projects" at Indiana University. I thank Professor Gregor von Laszewski and associate instructors for their support throughout the course.

REFERENCES

[1] Aerospike, "What is a key-value store?" WebPage. [Online]. Available: http://www.aerospike.com/what-is-a-key-value-store

[2] K. B. Margo Seltzer, "Berkeley db," WebPage, Aug. 2012. [Online]. Available: http://www.aosabook.org/en/bdb.html

[3] H. Chu, "Lightning memory-mapped database," WebPage, Nov. 2011. [Online]. Available: https://en.wikipedia.org/wiki/Lightning_Memory-Mapped_Database

[4] WhatIs, "L1 and l2," WebPage, Apr. 2005. [Online]. Available: http://whatis.techtarget.com/definition/L1-and-L2

[5] Wikipedia, "Multics," WebPage, Feb. 2017. [Online]. Available: https://en.wikipedia.org/wiki/Multics

[6] J. Mayo, "btest.c," WebPage, Jan. 2012, originary written by Martin Hedenfalk. [Online]. Available: https://github.com/OrangeTide/btree/blob/master/btest.c

[7] Symas Corporation, "Wrappers for other languages," WebPage. [Online]. Available: https://symas.com/offerings/lightning-memory-mapped-database/wrappers/

[8] H. Chu, "Lightning memory-mapped database manager (lmdb)," Dec. 2015. [Online]. Available: http://www.lmdb.tech/doc/

[9] D. Hellmann, "sqlite3 – embedded relational database," WebPage, Jan. 2017. [Online]. Available: https://pymotw.com/2/sqlite3/

[10] Basho Technologies, Inc., "Leveldb," WebPage. [Online]. Available: http://docs.basho.com/riak/kv/2.2.1/setup/planning/backend/leveldb/

[11] Wikipedia, "Leveldb," WebPage, Mar. 2017. [Online]. Available: https://en.wikipedia.org/wiki/LevelDB

[12] fallabs, "Kyoto cabinet: a straightforward implementation of dbm," WebPage, Mar. 2011. [Online]. Available: http://fallabs.com/kyotocabinet/

[13] B. Desmond, "Second strike with lightning!" WebPage, May 2013. [Online]. Available: http://www.anchor.com.au/blog/2013/05/second-strike-with-lightning/

[14] H. Chu, "Postfix lmdb adapter," WebPage. [Online]. Available: http://www.postfix.org/lmdb_table.5.html

[15] Symas Corp, "Database microbenchmarks," WebPage, Sep. 2012. [Online]. Available: http://www.lmdb.tech/bench/microbench/

SciDB: An Array Database

PIYUSH RAI[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: piyurai@iu.edu*
[+] *HID - S17-IO-3014*

April 30, 2017

SciDB is an open-source array database developed and maintained by Paradigm4. This paper explores it's archtectural overview and it's features designed to optimize the array data processing.

Keywords: Array Database, SciDB

https://github.com/piyurai/sp17-i524/tree/master/paper2/S17-IO-3014/report.pdf

1. INTRODUCTION

SciDB is an open source DBMS based on multi-dimensional array data model and runs on Linux platform. The data store is optimized for mathematical operations such as linear algebra and statistical analysis. The data is organized into arrays which can be stored in different forms including vectors and matrices with general form being n-dimensional arrays. It can be used to used to analyze scientific data from various data acquisition devices like sensors and wearables [1]. Array databases are optimized for homogeneous data models which are found in the field such as IOT and earth and space observation.

2. ARCHITECTURAL OVERVIEW

The array and their dimensions are named. The data tuple stored in each cell contain values for one or more attributes which are named and typed. The operations such as to filter and join arrays and aggregation over the cell values are supported. The arrays can be partitioned and per-partition values such as sum can be computed. Operations on large sized arrays could be performed without requiring them to fit in-memory. It also has 'missing data' facilities to reduce noise and impute missing values in the data [1].

The arrays are divided into chunks and partitioned across the nodes in the cluster, with provision of caching some of them in the main memory. SciDB uses shared-nothing architecture. Each node in the cluster has it's own resources such as memory, storage and CPU. The nodes can be added to or removed from the cluster without stopping the system. SciDB is ACID compliant. It uses Multiversion Concurrency Control (MVCC) to retain old values of data and maintains log of the changes in database state over time. In case the data history isn't required, the facility of purging the redundant data is also available.

Scalability and parallelism are the core requirements behind the design philosophy. Multiple machines connected over a network cluster are running server software instances. Disjoint subsets of data are stored locally on each machine. The input query is divided into smaller components which is applied by each instance to it's locally stored data. The information regarding which data is stored at what node is maintained by using Multidimensional Array Clustering (MAC) [2]. Data values close to each other are stored in the same chunk of storage media for faster access of range selects. It helps to minimize data movements among nodes. An optional user-settable chunk overlap can also be employed which enhances the performance of window queries that straddle two chunks. SciDB maintains a liveness-list of functioning instances which keeps getting updated as and when the nodes are added or removed from the cluster.

3. EXTENSIBILITY

To be able to cover different applications for scientific computations, the existing computing functionality can be extended by embedding user-defined functions and algorithms into the system. It uses registries to maintain information about data such as types and functions available in the system. This information is used to break down the query into sequence of operations. A number of client APIs are provided so that the system can be interfaced with tools like R and Python. With SciDB-R and SciDB-Py libraries, data in SciDB can be referenced as data frame objects.Bulk and parallel loading is supported for various formats such as binary and text.

4. ALTERNATIVES

SciDB tries to overcome the problem of scalability with traditional RDBMS and and lack of ACID support in NOSQL databases [3]. There are a few other array databases such as Rasdaman database. Rasdaman is one of the oldest array databases to be rolled out [4]. It's open source and it's source code is easily available. It's also available in form of VM image. However,

it's source code is divided into two communities, one open-source and other enterprise. SciDB provides Amazon Machine Images to try out the database. SciDB project also started as an open source project. However, instructions and source code for installing SciDB are available at [5] which can be accessed only after registration. SciDB has an in-depth documentation available explaining it's core features and design.

5. CONCLUSION

SciDB is designed for scalability and performance while at the same time complying with ACID properties. It also employs multiple methodologies to enhance the performance of the queries by dividing the data across multiple nodes and to reduce the data flow among nodes. It also provides the facility to embed user defined functions and algorithms into the system so that large variety of big data applications requiring scientific computation on homogeneous collection of data can benefit from it. SciDB is developed and supported by Paradigm4.

REFERENCES

[1] Paradigm4, "Architectural summary and motivation for paradigm4's scidb," Paradigm4, Whitepaper. [Online]. Available: http://discover.paradigm4.com/scidb-database-for-21st-century.html

[2] "Multidimensional array clustering," Web Page, 2017.

[3] J. Vaughan, "High programming overhead for nosql," blog, dec 2011. [Online]. Available: http://searchmicroservices.techtarget.com/blog/Microservices-Matters/Stonebraker-sees-high-programming-overhead-for-NoSQL

[4] "Rasdaman - impact and uses," Web Page.

[5] A. Poliakov, "Index of scidb releases," Web Page, mar 2015. [Online]. Available: http://forum.paradigm4.com/t/index-of-scidb-releases/773

Cassandra

SABYASACHI ROY CHOUDHURY[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: sabyasachi087@gmail.com

project-000, April 30, 2017

Apache Cassandra is a 'NoSql' database meant to handle a large volume of data through use of commodity hardwares. In this paper we examine Cassandra by understanding the architecture and internal data flows.

Keywords: Cloud, I524

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IO-3015/report.pdf

1. INTRODUCTION

Apache Cassandra is an open source column-oriented database that bases its distribution design on Amazon's Dynamo and its data model on Google's Bigtable [**?**]cassandra-book. It was developed by Facebook to handle large volume of writes and fault tolerance. The choice of database is solely on the basis of requirements. Cassandra is meant for scalability. If the need is to support thousands of write operations with millions of records, Cassandra or any other column oriented database is much more suitable. But if your need is to support transactions and considerably lower read/write access, RDBMS (Relational Database Management System) is best suited.

1.1. Column Oriented Database

Column Oriented Database 'cite:'www-column-db uses columns to store data tables rather than using rows as in traditional RDBMS. The main difference between column and row approach is schema definition. Column oriented databases have flexibility in column, in which it is not necessary for all the rows have same columns structure, but in RDBMS they are fixed and same for all the rows.

2. TERMINOLOGIES

Partitioning [1]Cassandra is a distributed system where data is distributed across multiple nodes. Each node is responsible for a part of the data.

Replication [1]In a distributed architecture, if one node is down , one of the data source is also sacrificed along with it. To avoid this, data in each node is copied to multiple nodes ensuring fault truculence and resulting in no single point of failure.

Gossip Protocol [2]Since Cassandra is a distributed system, it is important for individual nodes to know the existence and state of each other. To do so , Cassandra uses Gossip protocol.

Memtable [1]It is an In-Memory-Table or a write back cache in which data has not yet flushed into disk.

Column Family [3]It is a NoSql object to store key-value pair. Each column family has one key mapped with set of columns. Each column contains column name , its value and timestamp.

Bloom Filters [1]This algorithm helps to determine of a key is not present in a specific location. This helps in reducing I/O operations.

3. ARCHITECTURE

3.1. Cassandra Cluster/Ring

We have already covered that Cassandra is a distributed system. Each node of the system is assigned with an id or name to uniquely identify it. The set of nodes which helps Cassandra to start up, are know as seeds. Cassandra uses this seed to retrieve informations about other existing nodes. It uses gossip protocol for intra node communication and failure detection. A node exchanges state informations only to other three nodes. This state information contains data about itself and about other known three members reducing IO operations.

3.2. Data Distribution and Replication

Each node of Cassandra, is responsible for a specific range or set of data. During the start-up, every node is assigned with range of token ensuring evenly distribution of data. In figure 1, 0-255 range of data is distributed in four nodes. Hashing technique reviewed earlier is use to create the token of the row key. The row key falling under any of the above shown range will be assigned to its corresponding node. Say for example if

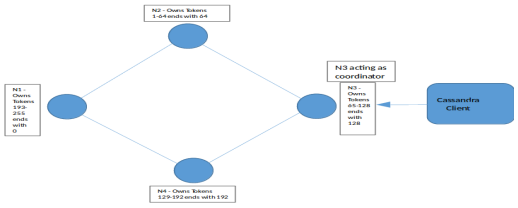


Fig. 1. Cassandra Cluster Ring

the hash value of the row key comes to 38 , it will go to the node N2. In a distributed system, once can't rely on a single node for storing a set of data, as if the node is down that particular set won't be available for read, write and update. To achieve a better reliability and fault tolerance , Cassandra replicates data in multiple nodes. It has two basic replication strategy :

1. Simple - In this data is copied on to the next node in a clockwise manner
2. Network-Topology - In this Cassandra is aware of the node's

3.3. Read and Write Paths

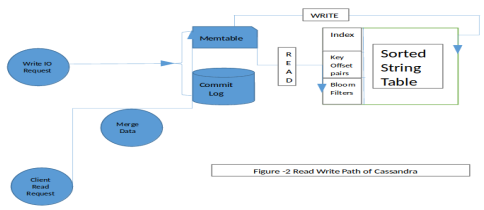


Fig. 2. Cassandra Read Write Data Flow

In figure 1, the client is connected with node 3. Since Cassandra is master less, N3 will be acting as coordinator and will serve for all client requests. It is the responsibility of N3 to communicate with its fellow nodes and fetch the desired results. We have already discussed that not all nodes communicates will all others for data retrieval instead it communicates only with handful of nodes for reducing IO operations. The number of nodes to communicate can be configured using QUORUM or knows as consistency level. Read and write flows has been described in figure 2. Each node processes write requests separately. It writes the data to Commit log first and then to Memtable. In case the node crashes, data can be restored from the Commit log. The data from Memtable will only be flushed to the disk or SSTable if

1. It reaches its maximum allocated size in memory
2. The number of minutes a memtable can stay in memory elapses.
3. Manually flushed by a user

Read operation is similar to write operation. Every read operation must be with row key. As discussed earlier, row-key is used to determine the right node and the request is then passed to that particular node. Read is then catered by the bloom-filter and then proceeds to the desired result set.

4. CHOOSING THE RIGHT DATABASE

4.1. Cap Theorem

Cap Theorem [4] states that, it is impossible for a distributed computer system to simultaneously provide more than two out of three of the following guarantees -

1. Consistency - Every read receives the most recent write or an error.
2. Availability - Every request receives a (non-error) response but doesn't guarantee if the data is recent or not.
3. Partition tolerance - System continues to operate even after loosing (dropped or delayed) an arbitrary number of messages by the nodes.

4.2. Cassandra and Cap Theorem

All Big Data databases are tolerant so considering Cap Theorem one have to choose between Consistency and Availability as per the need. Cassandra is highly available trading off with consistency. As we have seen so far, Cassandra master less architecture allows client to connect any of the node for read write. This increases availability as in case a node is down, the client can quickly jump to nearest available. The disadvantage is while reading data, the data might not be the recent. If the node with recent data us down, the data won't be replicated and reads will lead to the older state. Cassandra is used in write once and read many scenario. For example historical data analysis where the updates are very few.

4.3. Brief Comparison

Apart from Cassandra, Hbase and MongoDB are fairly popular choice of database for big data solutions. Hbase is built on HDFS (Hadoop Distributed File System) and column oriented database similar to Cassandra. Major advantage of Hbase is its querying functionality which is an edge over Cassandra. Hbase can also be installed on already clustered Hadoop environment. Check here for details. Accumulo is another option available which is based upon HDFS as well. The advantage of Accumulo is its cell level security where as all other column oriented databases have column level security. Having security at cell level allows user to see different value as appropriate based on the row. Check here for details. MongoDB stands separate from all of the aforesaid

mentioned as it is a document oriented database. Since data fields are to be stored vary between different elements, column oriented storage leads to lot of empty column values. MongoDB provides a way to store only the necessary fields. Check here for more details.

5. CONCLUSION

Cassandra is a column oriented database with focus on high availability. The architecture allows it for fast write operations. Cassandra can handle bulk writes and make it an ideal candidate for storing logs as logs are generally high in volume and speed. Cassandra is poor in table joins and hence not suited for data mining.

ACKNOWLEDGMENTS

I would like to thanks Akhil Mehra for his vibrant description of Cassandra architecture.

REFERENCES

[1] Akhil Mehera / Dzone, "Introduction to cassandra," Web Page, Apr. 2015, accessed: 2015-04-06. [Online]. Available: https://dzone.com/articles/introduction-apache-cassandras

[2] Wikipedia, "Gossip protocol," Web Page, Jan. 2017, accessed: 2017-01-11. [Online]. Available: https://en.wikipedia.org/wiki/Gossip_protocol

[3] Wikipedia, "Column family," Web Page, Jan. 2017, accessed: 2017-01-11. [Online]. Available: https://en.wikipedia.org/wiki/Column_family

[4] Wikipedia, "Cap theorem," Web Page, Jan. 2017, accessed: 2017-04-17. [Online]. Available: https://en.wikipedia.org/wiki/CAP_theorem

Apache Derby

RIBKA RUFAEL[1,*,+]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: rrufael@umail.iu.edu*
[+] *HID: S17-IO-3016*

paper2, April 30, 2017

Apache Derby, part of Apache DB subproject, is Java based relational database management system. Apache Derby database provides storage, access and secure management of data for Java based applications. Apache Derby is an open source software and it is licensed under Apache version 2.0. Apache Derby is written in Java and it runs on any certified JVM(Java Virtual Machine). JDBC driver in embedded or netwrok server frameworks allows applications to access Apache Derby Database.

Keywords: Apache Derby, relational database management system, JDBC

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IO-3016/report.pdf

1. INTRODUCTION

Apache Derby is an open source relational database management system(RDBMS). Applications interact with Apache Derby database through JDBC(Java Database Connectivity) driver. Apache Derby is built in Java programming language which makes it platform independent. Apache Derby can be embedded into Java based application or it can be setup to run in a client/server environment. Apache Derby complies with JDBC and ANSI SQL standards. Apache Derby database allows applications to create, update, read, delete and manage data [1, 2].

Apache Derby is part of the Apache DB subproject of Apache Software Foundation. It was in August 2004 that IBM submitted the Derby code to Apache Software Foundation. Then Apache Derby became part of the Apache DB project in July 2005 [2].

2. ARCHITECTURE

There are two views that describe Apache Derby's architecture. These views are Module View and Layer/Box view [3].

2.1. Module View

Modules and Monitor are components in Apache Derby database system. A collection of distinct functionality is a module. Examples of modules in Apache Derby are lock management, error logging and JDBC driver. Lock management is responsible for controlling concurrent transactions on data objects. Once Apache Derby database is up and running any informational messages or error messages are logged. This message logging is handled by an error logging module. Applications use JDBC driver to interact with Apache Derby database. A number of classes are used for the implementation of each module [3].

The monitor is responsible for managing Apache Derby database. Whenever a request to modules come, the monitor is responsible for selecting appropriate module implementation depending on what the module request was and from which environment the request came from [3].

2.2. Layer/Box View

JDBC, SQL, Store and Services are the four layers in Apache Derby. The Java Database Connectivity abbreviated JDBC is an API(Application Programming Interface) which allows applications to connect to Apache Derby database. JDBC interfaces that Apache Derby implements allow applications to connect to Apache Derby. Some of the interfaces are Driver, DataSource, ConnectionPoolDataSource, XADataSource and PreparedStatement. Apache Derby JDBC has implementation of java.sql and javax.sql classes [3].

The other layer which sits below JDBC is SQL layer. Compilation and execution are the two logical parts of SQL layer. SQL statement that is invoked by an application to Apache Derby Database passes through five step compilation process. First statement is parsed with a parser created by JavaCC(Java Compiler Compiler) and tree of query nodes is created. Second step is binding to resolve objects. Third step is optimizing to identify the access path. In fourth step Java class is created for the statement this is then cached to be used by other connections. Finally, on the fifth step, class is loaded and instance of generated statement is created. During execution, execute methods on the class instance created during compilation are called and Derby result set is returned. JDBC layer is responsible in converting the Derby result set into JDBC result set for the applications [3].

Access and raw are the two parts of the store layer. Raw data storage for data in files and pages, transaction logging

and management is handled by the raw store. The access store interfaces with SQL layer and takes care of scanning of tables and indexes, indexing, sorting, locking and etc [3].

Lock management, error logging and cache management are part of the service layer. Clock based algorithm is used by cache management. It is mainly used for caching buffer, caching compiled java classes for SQL statement implementation plans and caching table descriptors [3].

2.3. Shell Access

Shell access to Apache Derby database is achieved by ij. The ij tool is one of java utility tools that allows performing sql scripts on Derby database in embedded or network server frameworks. The ij tool can be used for creating database, connecting to database, run and execute sql scripts. Commands in ij are case sensitive and semicolon is used to mark end of command. Other utility tools that come with Apache Derby are sysinfo , dblook and SignatureChecker. The sysinfo utility tool is used to get version and other information about Apache Derby and the Java environment. The dblook utility is used to generate Data Definition Language(DDL) for Derby database. Checks, functions, indexes, jar files, primary keys, foreign keys and schemas are the objects generated by dblook. SignatureChecker is used to check whether functions and procedures used in Derby database comply with the rules and standards [4].

2.4. API

Applications can interact with Apache Derby database through two JDBC drivers org.apache.derby.jdbc.EmbeddedDriver and org.apache.derby.jdbc.ClientDriver for embedded and network server frameworks respectively [4].

3. INSTALLATION

Java 2 Standard Edition (J2SE) 6 or higher is needed for installing Apache Derby and for running Derby the Java Runtime Environment (JRE) is needed. Apache Derby can be downloaded from the Apache DB download page [5]. Apache Derby can be installed on Windows, MAC, UNIX and Linux operating systems. After downloading the zipped installation file, the file should be extracted into directory and then DERBY_INSTALL variable should be set in the same directory as where Derby was installed. Apache Derby provides two frameworks Embedded Derby and Derby Network Server [6].

3.1. Embedded Derby

In the embedded Derby framework, Apache Derby engine and the application which accesses it run on the same JVM(Java Virtual Machine). Embedded Derby JDBC driver is used by the application to interact with Apache Derby database. To setup Embedded Derby mode, derby.jar and derbytools.jar must be included in the CLASSPATH after installation. The Derby engine and Embedded Derby JDBC driver are included in derby.jar. derbytools.jar includes ij tool(utility tool which can be used as scripting tool to interact with Derby database). In Embedded Derby framework, multiple users that are running in the same JVM can access the same database. Figure 1 shows the Embedded Derby framework [6].

3.2. Derby Network Server

Derby Network Server framework allows multiple application running in the same JVM or different JVMs to access Derby

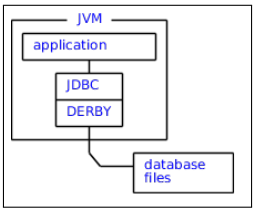


Fig. 1. Embedded Derby Framework [6].

database over the netwrok in a typical client/server architecture. In this framework application accesses Derby database through Derby Network Client JDBC driver. To setup Derby netwrok server derbynet.jar and derbytools.jar must be included in CLASSPATH on the server side. The program for Network Server and reference to the Derby engine are included derbynet.jar file. On the client side, derbyclient.jar and derbytools.jar must be included in CLASSPATH. Derby Network Client JDBC driver is included in derbyclient.jar file. Derby Network Server listens and accepts requests on port 1527 by default but this can be changed to a different port if needed. Figure 2 shows the Derby Network Server framework [6].

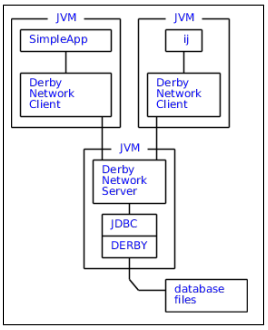


Fig. 2. Derby Network Server Framework [6].

Another variation for setting up Derby Network Server is embedded server. In embedded server, application will have both Embedded Derby JDBC driver and Network Server. The application uses Embedded Derby JDBC driver which runs on the same JVM and extends access to other applications running on different JVMs through the Network Server [6].

4. SECURITY

Apache Derby provides a number of security options. Some of these features are authentication, authorization and disk encryption. Before users are granted access to the Derby database, Derby can be setup to perform user authentication. There may be a need for certain user groups to have read only access to Derby database and some other user groups to have both read and write access to Derby database which can be achieved by Derby's user authorization feature. Data which is saved on disk can be encrypted with Derby's disk encryption feature [7].

5. USE CASES

Apache Hive, data warehouse querying and analysis software, uses Apache Derby database by default for metadata store. Apache Derby can be configured in the embedded or netwrok server mode for Hive meta data storage [8, 9].

My Money, management and analysis software for personal and business finances which is built by MTH Software, Inc, uses Apache Derby as a relational database management system [10, 11].

6. LICENSING

Apache Derby is an open source technology hence it is available for free. Apache Derby is licensed under Apache License, Version 2.0 [1].

7. EDUCATIONAL MATERIAL

Apache Derby tutorial page is one of the resources which can be used by users who are new to Apache Derby. This tutorial contains step by step information about Apache Derby installation, embedded framework configuration and network server framework configuration [6]. Derby Engine Architecture Overview page provides information about Apache Derby architecture [3]. Apache Derby documentation page has list of documentations that can be used as reference for new users, developers and administrators [12].

8. CONCLUSION

Apache Derby database offers storage,access and secure management of data for Java based applications. Apache Derby provides complete relational database management package that complies with JDBC and ANSI SQL standards. Apache Derby's small size makes it suitable for embedding it into applications which run on smaller devices with less physical memory. As the use case of Hive using Apache Derby for metadata storage indicates, Apache Derby can be incorporated into software stack for big data projects for metadata storage.

ACKNOWLEDGEMENTS

The author would like to thank Professor Gregor von Laszewski and associate instructors for their help and guidance.

REFERENCES

[1] Apache Software Foundation, "Apache Derby," Web Page, Oct. 2016, accessed: 2017-03-14. [Online]. Available: https://db.apache.org/derby/

[2] The Apache Software Foundation, "Apache Derby Project Charter," Web page, Sep. 2016, accessed: 2017-02-25. [Online]. Available: https://db.apache.org/derby/derby_charter.html

[3] Apache Software Foundation, "Derby Engine Architecture Overview," Web Page, Sep. 2016, accessed: 2017-03-14. [Online]. Available: http://db.apache.org/derby/papers/derby_arch.html#Module+View

[4] Apache Software Foundation, "Derby Tools and Utilities Guide," Web Page, Oct. 2016, accessed: 2017-03-24. [Online]. Available: https://db.apache.org/derby/docs/10.13/tools/derbytools.pdf

[5] Apache Software Foundation, "Apache Derby: Downloads," Web Page, Sep. 2016, accessed: 2017-03-24. [Online]. Available: http://db.apache.org/derby/derby_downloads.html

[6] Apache Software Foundation, "Apache Derby Tutorial," Web Page, Sep. 2016, accessed: 2017-03-14. [Online]. Available: https://db.apache.org/derby/papers/DerbyTut/index.html

[7] Apache Software Foundation, "Derby and Security," Web Page, Mar. 2017, accessed: 2017-03-20. [Online]. Available: http://db.apache.org/derby/docs/10.8/devguide/cdevcsecuree.html

[8] Apache Software Foundation, "Apache Hive," Web Page, Mar. 2017, accessed: 2017-03-23. [Online]. Available: https://cwiki.apache.org/confluence/display/Hive/Home

[9] Apache Software Foundation, "GettingStarted-Metadata Store," Web Page, Mar. 2017, accessed: 2017-03-23. [Online]. Available: https://cwiki.apache.org/confluence/display/Hive/GettingStarted#GettingStarted-MetadataStore

[10] MTH Software, Inc, "MTH built products," Web Page, Mar. 2017, accessed: 2017-03-23. [Online]. Available: http://www.mthbuilt.com/products.php

[11] MTH Software, Inc, "How to use SQL in My Money," Web Page, Mar. 2017, accessed: 2017-03-23. [Online]. Available: http://wiki.mthbuilt.com/How_to_use_SQL_in_My_Money

[12] Apache Software Foundation, "Apache Derby: Documentation," Web Page, Oct. 2016, accessed: 2017-03-24. [Online]. Available: https://db.apache.org/derby/manuals/index.html

Facebook Tao

NANDITA SATHE[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding author: nsathe@iu.edu*

paper-2, April 30, 2017

Facebook Tao is a graph database currently serving Facebook Inc. to manage data of its billions of users. Graph database stores data in a graph structure and establishes relationships between data using nodes and edges. Graph databases are ideal for systems that require data to be represented as a graph or hierarchical structure and need to establish connections between data points. Storing relationships between data points as a first class entity helps to draw insights from the data. For example, real time recommendation engine is possible because of the ability of graph database to connect to the masses of buyers to product data, thereby enabling insights on customer needs and product trends. A graph database system like Tao is apt for relationship driven data requirements of Facebook. Main aim of Tao system is to achieve lowest read latency, timeliness in writes and efficiently scaling the data.

Keywords: Facebook Tao, Graph Database, memcache

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IO-3017/report.pdf

1. INTRODUCTION

Graph databases like Facebook Tao are a response to data needs that traditional RDBMSs like MySQL do not meet. For example, Facebook realized however efficient relational database it would use, it is not sufficient to manage the enormous data challenge Facebook had. The data was a social graph. Another mismatch, which relational database or block cache had was, most of the data that would be read into cache did not belong to any relation. For example, 'If user likes that picture". In most records the answer would be 'No' or 'False'. Storing and reading this unwanted data was a burden. Meanwhile Facebook users' base was increasing daily. Ultimately Facebook came up with Facebook Tao, a distributed social graph data store.

Facebook TAO (The Association and Objects) is a geographically distributed data store that provides timely access to the social graph for Facebook's demanding workload using a fixed set of queries [1]. It is deployed at Facebook for many data types that fit its model. The system runs on thousands of machines, is widely distributed, and provides access to many petabytes of data. TAO represents social data items as Objects (user) and relationship between them as Associations (liked by, friend of). TAO cleanly separates the caching tiers from the persistent data store allowing each of them to be scaled independently. To any user of the system it presents a single unified API that makes the entire system appear like 1 giant graph database [2]. Key advantages of the system include [2]:

- Provides a clean separation of application/product logic from data access by providing a graph API and data model to store and fetch data.
- By implementing a write-through cache TAO allows Facebook to provide a better user experience and preserve the all important read-what-you-write consistency semantics even when the architecture spans multiple geographical regions.
- By implementing a read-through write-through cache TAO also protects the underlying persistent stores better by avoiding issues like thundering herds without compromising data consistency.

2. TAO'S GOAL

Main goal of implementing Tao is efficiently scaling the data. Facebook handles approximately a billion requests per second [3]. So obviously data store has to be scalable. More than that, scalability should be efficient otherwise scaling data across machines would be extremely costly.

Second goal is to achieve lowest possible read latency. So that if a user has commented on a post, the original post writer should be able to read it immediately. Efficiency in Scaling and low Read latency is achieved by (i) separating cache and data storage, (ii) Graph specific caching and (iii) Sub-dividing data centers [3].

Third goal is to achieve timeliness of writes. If a web server has written something and it sends a read request, it should be able to read the post. Write timeliness is achieved by (i) Write through cache and (ii) Asynchronous replication [3].

Lastly, the goal is also to have high read availability which is achieved by using alternate data sources.

3. TAO DATA MODEL AND API

Facebook Inc. explains TAO data model and API associated with using an example [4]. Figure 1 depicts TAO data model.

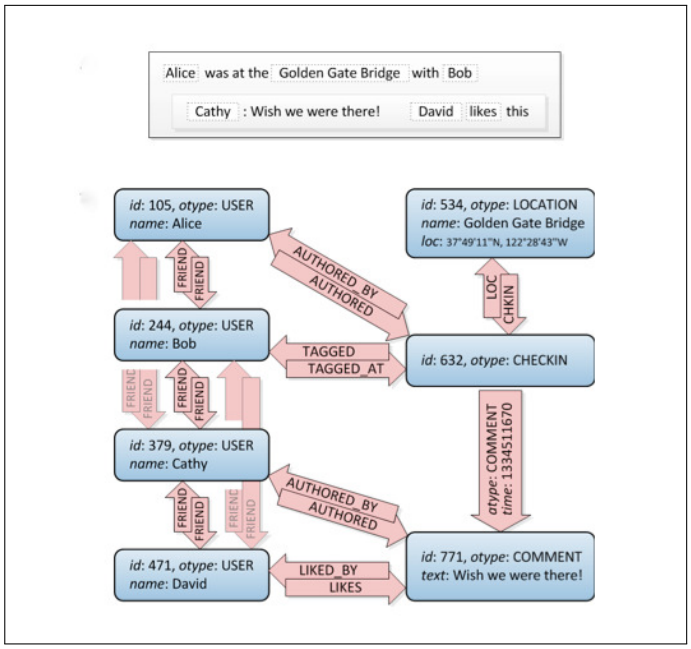


Fig. 1. TAO Data Model [4].

This example shows a subgraph of objects and associations that is created in TAO after Alice checks in at the Golden Gate Bridge and tags Bob there, while Cathy comments on the check-in and David likes it. Every data item, such as a user, check-in, or comment, is represented by a typed object containing a dictionary of named fields. Relationships between objects, such as "liked by" or "friend of," are represented by typed edges (associations) grouped in association lists by their origin. Multiple associations may connect the same pair of objects as long as the types of all those associations are distinct. Together objects and associations form a labeled directed multigraph.

For every association type a so-called inverse type can be specified. Whenever an edge of the direct type is created or deleted between objects with unique IDs *i*d1 and *i*d2, TAO will automatically create or delete an edge of the corresponding inverse type in the opposite direction (*i*d2 to *i*d1). The intent is to help the application programmer maintain referential integrity for relationships that are naturally mutual, like friendship, or where support for graph traversal in both directions is performance critical, as for example in "likes" and "liked by."

3.1. Objects and Associations

TAO objects are typed nodes, and TAO associations are typed directed edges between objects. Objects are identified by a 64-bit integer (id) that is unique across all objects, regardless of object type (otype). Associations are identified by the source object (*i*d1), association type (atype) and destination object (*i*d2). At most one association of a given type can exist between any two objects. Both objects and associations may contain data as key→value pairs. A per-type schema lists the possible keys, the value type, and a default value. Each association has a 32-bit time field, which plays a central role in queries [1].

Object: (id) -> (otype, (key -> value)⋆)

Assoc.: (id1, atype, id2) -> (time, (key -> value)⋆)

Figure 'Tao Data Model' shows how TAO objects and associations might encode the example, with some data and times omitted for clarity. The example's users are represented by objects, as are the checkin, the landmark, and Cathy's comment. Associations capture the users' friendships, authorship of the checkin and comment, and the binding between the checkin and its location and comments.

The set of operations on objects is of the fairly common create/set-fields/get/delete variety. All objects of a given type have the same set of fields. New fields can be registered for an object type at any time and existing fields can be marked deprecated by editing that type's schema.

Associations are created and deleted as individual edges. If the association type has an inverse type defined, an inverse edge is created automatically. The API helps the data store exploit the creation-time locality of workload by requiring every association to have a special time attribute that is commonly used to represent the creation time of association. TAO uses the association time value to optimize the working set in cache and to improve hit rate [1].

4. TAO ARCHITECTURE

TAO is separated into layers: two caching layers and a storage layer.

4.1. Storage Layer

The data is persisted using MySQL. The API is mapped to a small number of SQL queries. TAO needs to handle a far larger volume of data than can be stored on a single MySQL server and so, the data is divided into logical shards. Each shard is contained in a logical database. Database servers are responsible for one or more shards. The number of shards far exceeds the number of servers. The shard to server mapping is tuned in to balance load across different hosts. By default all object types are stored in one table, and all association types in another. Every 'object-id' has a corresponding 'shard-id'. Objects are bounded to a single shard throughout their lifetime. An association is stored on the shard of its *i*d1, so that every association query can be served from a single server. [3].

4.2. Caching Layer

TAO's cache implements the API for clients, handling all communication with databases. A region/tier is made of multiple closely located Data centers. Multiple Cache Servers make up a tier (set of databases in a region are also called a tier) that can collectively capable of answering any TAO Request. Each cache request maps to a server based on sharding. Write operations on an association with an inverse may involve two shards, since the forward edge is stored on the shard for *i*d1 and the inverse edge is on the shard for *i*d2. Handling writes with multiple shards involve: Issuing an RPC (Remote Procedure Call) call to the member hosting *i*d2, which will contact the database to create the inverse association. Once the inverse write is complete, the caching server issues a write to the database for *i*d1. TAO does not provide atomicity between the two updates. If a failure occurs the forward may exist without an inverse, these hanging associations are scheduled for repair by an asynchronous job [3].

4.3. Leaders and Followers

There are two tiers of caching clusters in each geographical region. Clients talk to the first tier, called followers. If a cache miss occurs on the follower, the follower attempts to fill its cache from a second tier, called a leader. Leaders talk directly to a MySQL cluster in that region. All TAO writes go through followers to leaders. Caches are updated as the reply to a successful write propagates back down the chain of clusters. Leaders are responsible for maintaining cache consistency within a region. They also act as secondary caches [4].

4.4. Scaling Geographically

High read workload scales with total number of follower servers. The assumption is that latency between followers and leaders is low. Followers behave identically in all regions, forwarding read misses and writes to the local region's leader tier. Leaders query the local region's database regardless of whether it is the master or slave. This means that read latency is independent of inter-region latency. Writes are forwarded by the local leader to the leader that is in the region with the master database. Read misses by followers are 25X as frequent as writes in the workload thus read misses are served locally [3]. Facebook chooses data center locations that are clustered into only a few regions, where the intra-region latency is small (typically less than 1 millisecond) [3]. It is then sufficient to store one complete copy of the social graph per region.

Since each cache hosts multiple shards, a server may be both a master and a slave at the same time. It is preferred to locate all of the master databases in a single region. When an inverse association is mastered in a different region, TAO must traverse an extra inter-region link to forward the inverse write. TAO embeds invalidation and refill messages in the database replication stream. These messages are delivered in a region immediately after a transaction has been replicated to a slave database. Delivering such messages earlier would create cache inconsistencies, as reading from the local database would provide stale data. If a forwarded write is successful then the local leader will update its cache with the fresh value, even though the local slave database probably has not yet been updated by the asynchronous replication stream. In this case followers will receive two invalidates or refills from the write, one that is sent when the write succeeds and one that is sent when the write's transaction is replicated to the local slave database [3].

5. EDUCATIONAL MATERIAL

To get started on learning Facebook TAO, following resources can prove helpful.

- Technical paper on Facebook TAO [1].
- Background, Architecture and Implementation from Facebook itself [4].
- TAO summary in a video on USENIX website [5].

6. RELATED WORK

TAO is a geographically distributed, eventually consistent graph store optimized for reads, thus combining all three techniques into one system. Eventually consistency model is based on BASE (Basically Available, Soft state, Eventual consistency) semantics. These systems typically provide weaker guarantees than the traditional ACID (Atomicity, Consistency, Isolation, Durability) semantics. Google's BigTable, Yahoo!'s PNUTS, Amazon's SimpleDB, and Apache's HBase are examples of this more scalable approach. These systems all provide consistency and transactions at the per-record or row level similar to TAO's semantics for objects and associations, but do not provide TAO's read efficiency or graph semantics [1].

The Coda file system uses data replication to improve performance and availability in case of unreliable networks. Unlike Coda, TAO does not allow writes in portions of the system that are disconnected. Google Megastore is a storage system that uses Paxos (protocol for distributed consensus) across geographically distributed data centers to provide strong consistency guarantees and high availability. TAO provides no consistency guarantees but handles comparatively many more requests [1].

Since TAO was designed specifically to serve the social graph, its features are inherited from the graph databases. Neo4j is a open-source graph database that provides ACID semantics and the ability to shard data across several machines. Twitter uses its FlockDB to store parts of its social graph, as well [1].

Instead of using existing graph systems Facebook has customised Tao to fulfill its specific needs and workload requirements.

7. CONCLUSION

Overall, this paper explains characteristics and challenges of Facebook's workload, the objects and associations data model and lastly, it details out TAO, the geographically distributed system that implements the API to work with social graph. TAO is deployed at scale inside Facebook. Its separation of cache and persistent store has allowed those layers to be independently designed, scaled, and operated, and maximizes the reuse of components across the organization [1].

8. ACKNOWLEDGEMENTS

The author would like to thank Prof. Gregor von Laszewski and his associates from the School of Informatics and Computing for providing all the technical support and assistance.

REFERENCES

[1] N. Bronson *et al.*, "Tao: Facebook's distributed data store for the social graph," in *2013 USENIX Annual Technical Conference*, 2013. [Online]. Available: http://ai2-s2-pdfs.s3.amazonaws.com/39ac/2e0fc4ec63753306f99e71e0f38133e58ead.pdf

[2] V. Venkataramani, "What is the tao cache used for at facebook," Web Page, June 2013. [Online]. Available: https://www.quora.com/What-is-the-TAO-cache-used-for-at-Facebook

[3] N. Upreti, "Facebook's tao and unicorn data storage and search platforms," Slides, April 2015. [Online]. Available: https://www.slideshare.net/nitishupreti/faceboko-tao-unicorn

[4] M. Marchukov, "Tao: The power of the graph," Web Page, June 2013. [Online]. Available: https://www.facebook.com/notes/facebook-engineering/tao-the-power-of-the-graph/10151525983993920/

[5] N. Bronson, "Tao: Facebook's distributed data store for the social graph," Slides, June 2013. [Online]. Available: https://www.usenix.org/node/174510

InCommon

MICHAEL SMITH[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: mls35@iu.edu

paper2, April 30, 2017

InCommon is a federated security service that is responsible for the management of identity verification solutions serving U.S. education and research. All users within this federation allow partners to share identity information in order easily recognize the user. This federation provides numerous benefits for users and service providers through the convenience of single sign on capabilities for the user. Privacy is enhanced by limiting the distribution of personal information amongst numerous service providers. Scalability is easily facilitated due to the unified policies and management procedures. Programs such as InCommon assurance and university case studies are examined.

Keywords: InCommon, User authentication, identity management, I524

ttps://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IO-3019/report.pdf

1. INTRODUCTION

Electronic credentialing of individuals requires an effective implementation of a set of policies and procedures. In order to be successful, identity management requires an organization to keep user information up to date, providing the trust needed for secure transactions, and determine user access of online applications. The major issues with identity management is the increasing number of cloud services or applications that are web hosted, all of which have different policies for implementing identity verification. The solution is to establish a federation which is defined as "an association of organizations that come together to exchange information, as appropriate, about their users and resources in order to enable collaborations and transaction" [1]. Within this federation the parties come into an accordance on the policies associated with identity management. A great example of a federation that encompasses this definition is InCommon.

2. INCOMMON

InCommon was founded by the advanced technology organization Internet2. Their mission is to create an environment that facilitates the ability for educators and researchers to collaborate regardless of their location. Their network encompasses over 90,000 institutions, 305 universities, 70 government agencies with network operations center powered by Indiana University [2]

Through the InCommon service, users will not have to remember a plethora of usernames and passwords for each web service. Instead, they will be able to have single sign on (SSO) conveniences. Giving time back to faculty, staff and students for education, research and other contributions to the University. Any service provider within this federation no longer needs to manage databases of username and passwords, the users are verified and then administered security tokens to then engage with service providers within the federation. By limiting the amount of identity information required of the service provider, the users privacy is safer in the event of a security breach of the service provider.

3. ARCHITECTURE

The architecture of the InCommon framework is comprised of several key components such as SAML, Shibboleth, certificate service offerings and Duo. While different from one another these components all help assist the federation with user authentication. SAML is the language used in the transfer of data, Shibboleth is a service that assists in the implementation of SAML. The certificate service mainly comprised of SSL assist in the privacy of data passed, and Duo provides an extra layer of security with two factor authentication. Each component will be discussed in the following sections.

4. SAML

The language used by InCommon is referred to as security assertion markup language or SAML. This language is based in XML which allows for the exchange of authentication information between a user and a provider [3]. It is the industry accepted standard language for identity verification by numerous government, businesses and service providers[4]. The general user verification is done by an identity provider(IdP) which is responsible for user authentication through the use of security tokens with SAML 2.0 [5]. Service providers (SP) are defined as entities

that provide web services, internet, web storage etc. They rely on the IDPs for the verification process. A significant amount of the major web service providers such as Google, Facebook, Yahoo, Microsoft, and Paypal play a dual role and exist also as identity providers.

5. SHIBBOLETH

Shibboleth is the service that has a suite of products that assist the InCommon federation through utilization of SAML in programming languages such as C ++ and Java[6]. The normal authentication process for Shibboleth is to intercept access to a service, determine who is the identity provider for the user. Once the identity provider has been discovered a SAML authentication request is sent to the identity provider. Identity providers SAML response will have the relevant user information for verification. The extracted user information will then be passed to the service provider or resource determining user accessibility. While the process sounds complex it will occur instantaneously, after the user has entered its single sign on.

6. CERTIFICATE SERVICE

The types of certificates that InCommon have available for issue are SSL/TLS, extended validation, client, code signing, IGTF server, and elliptical curve cryptography certificates (ECC). SSL (secure sockets layer) is "the standard security technology for establishing an encrypted link between a web server and a browser. The link ensures all data is passed between the web server and browers remain private and integral"[7]. The details of an SSL certificate issued by InCommon will contain user information and the expiration date. For all educational institutions, InCommon offers unlimited server and client certificates for the annual fee.

7. DUO

In collaboration with the trusted access company Duo, Incommon offers two factor authentication through the utilization of the users smart phone[8]. A duo mobile app supports the following platforms: Apple iOS, Google Android, Windows mobile, Palm WeboS, Symbian OS, RIM blackberry, Java J2ME. The application will generate a randomly generated one time password that the user will type into the web application for a more secure identity verification. Two factor authentication does not require smartphone, other methods such as automate voice calls or SMS messages. In addition to Duo mobile, a service called Duo push is available which does not require the user to type in the password, authentication occurs directly from the mobile app. It is up to the university to determine how Duo is deployed, whether it will occur with the identity provider or the service provider. If it is deployed at the service provider destination, Duo web supports the following client libraries: python, ruby, classic ASP, ASP.net, Java, PHP, Node.js, ColdFusion, and Perl.

8. ASSURANCE PROGRAM

Many organizations and government agencies such as a national institute of health and public universities are requiring identity providers to become certified in this program. Incommon offers an assurance program that will examine and the practices of an organization and will rank them based on a number of criteria. Areas of examination include "...identity proofing(such as checking government issued ID before accepting that people are who they say they are), password handling (including making sure that passwords are not sent or stored in the clear), and authentication(such as ensuring the resistance of an authentication method to session hijacking)" [9].

There are two levels of assurance in the InCommon program, bronze and silver. Bronze is comparable to NIST level of Assurance 1, which is for common usage of internet identity management. Silver is comparable to NIST level of Assurance 2, which defines the institute as having sufficent requirements provide a security at the level for a financial transaction [10]. NIST levels of security are set by the National Institute of Standards and Technology, a government body within the U.S. Department of Commerce[11]. Compliance with a bronze level only requires a level of self certification of the requirements, where as a silver level is more difficult to achieve. A third party or evaluator that has been verified by InCommon is required to peform an audit ensuring the identity provider is meeting all the rules and requirements.

9. UNIVERSITY OF MINNESOTA

One example where InCommon was successfully deployed was at the University of Minnesota. They joined the InCommon federation on September 2010 [12]. The university contains 51,000 students, and over 300 institutes. Their previous identity management vendor charged on a per certificate basis. This differs from InCommon which offers an annual fee with unlimited certificates. This simplifies the ability for IT departments within Universities to properly budget. Additionally the university saw a cost savings of 38,000 dollars. This model also encourages enhancing security because cost does not influence which servers to secure.

10. STUDENTS ONLY

The bottom line of a university is not the only one who sees the cost benefits of InCommon [13]. Student verification provider known as students only is a way for students to enroll to verify their status as a student. This verification is then passed to businesses that would like to offer discounts to students. To prevent nonstudents from taking advantage of offerings of companies it can be cumbersome for a student to properly verify their status. With the help of InCommon Students Only helped streamline the process for students to verify their identity in a single sign on. This reassured the companies and students were able to save money without the difficulties of personally handling identity verification.

11. CONCLUSION

As the number of services on the web continue to grow it can be quite challenging for both universities and service providers to properly manage accessibility manage identities. InCommon hopes to address this issue by bringing U.S. educational institutes into the same federation. This will create a common groundwork of policies and procedures related to identity management. Through this unity, users such as faculty, staff, and students alike can benefit from the obvious conveniences of single sign on. However, they will also benefit from enhanced security and privacy. Institutions that have entered into InCommon have seen benefits such as cost savings over competitors in this market as well as simplification of the billing process for University IT. The unlimited certificate model as well as the

diverse types of certificates allows IT flexibility to issue the appropriate certificate without the worry of budgeting constraints. Partners such as Duo further improve security through two factor authentication dramatically improving the protection of the user.

REFERENCES

[1] InCommon, "Incommon overview," Webpage. [Online]. Available: https://www.incommon.org/docs/presentations/InCommon_Overview.ppt

[2] InCommon, "What is the incommon federation?" Webpage. [Online]. Available: https://spaces.internet2.edu/download/attachments/2764/final_InCommon.pdf

[3] Wikipedia, "Security assertion markup language," Webpage. [Online]. Available: https://en.wikipedia.org/wiki/Security_Assertion_Markup_Language

[4] Pingidentity, "Saml: How it works," webpage. [Online]. Available: https://www.pingidentity.com/en/resources/articles/saml.html

[5] empowerID, "Service providers, identity providers & security token services," Webpage. [Online]. Available: https://www2.empowerid.com/learningcenter/technologies/service-identity-providers

[6] Shibboleth, "Shibboleth," Webpage. [Online]. Available: https://shibboleth.net/

[7] SSL, "What is ssl?" Webpage. [Online]. Available: http://info.ssl.com/article.aspx?id=10241

[8] InCommon, "Incommon multifactor," Webpage. [Online]. Available: https://www.incommon.org/duo/

[9] M. Erdos, "An introduction to assurance," Webpage. [Online]. Available: http://iam.harvard.edu/resources/introduction-assurance

[10] InCommon, "The incommon assurance program," Webpage. [Online]. Available: https://www.incommon.org/assurance/

[11] Nist, "Nist," Webpage. [Online]. Available: www.nist.gov

[12] InCommon, "The university of minnesota enables security at scale with incommon," Webpage, July 2016. [Online]. Available: https://www.incommon.org/docs/eg/InC-Cert_CaseStudy_Minnesota.pdf

[13] InCommon, "Incommon flies high with students only," Webpage. [Online]. Available: https://www.incommon.org/docs/eg/InC_CaseStudy_StudentsOnly_2009.pdf

Hadoop YARN

MILIND SURYAWANSHI[1,2] AND GREGOR VON LASZEWSKI[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[2] Savitribai Phule Pune University 2010, Pune, Maharashtra 411007 India
[*] Corresponding authors: laszewski@gmail.com

project-paper2, April 30, 2017

Apache Hadoop 2.0 came up with Yarn architecture which is capable of managing resources and processing of tasks separately. This is the most important implementation over MapReduce which increased the scalability of Hadoop 1.0. YARN is the next generation of Hadoop's compute platform.

Keywords: Hadoop, Yarn, MapReduce, Cluster

https://github.com/MilindSuryawanshi/sp17-i524/tree/master/paper2/S17-IO-3020/report.pdf

1. INTRODUCTION

Apache Hadoop [1] is open source software framework, can be installed on a cluster of computers, which can communicate together for large amount of data storage and processing it, in highly distributed manner. Hadoop 2.0 project was developed to resolve the limitations of MapReduce (we will see the limitation as we move ahead). YARN (Yet Another Resource Negotiator) is Apache Hadoop's cluster resource management system [2]. It's a resource management technology which makes pace between, the way applications use Hadoop system resources and node manager agents. To understand Yarn, it is important to know the limitation of Hadoop1.0 MapReduce, which enforced the creation of Hadoop 2.0. Hadoop 2.0 alpha version introduced in August 2013 [3].

2. LIMITATION OF MAPREDUCE

Hadoop 1.0 was designed for big data processing and MapReduce was the only option supported.MapReduce is the Hadoops framework, which Maps the multi terabyte data sets into chunks which will getting executed in parallel manner. The sorted output of the Map then given to Reduce tasks [4]. In MapReduce framework Node holds the meta data information and the daemon Job Tracker will be ensuring that the Task Tracker are processing the data. Task Tracker runs the Map and reduces the tasks and reports the status to Job Tracker. Job Tracker tracks the tasks, schedules the jobs and monitors the Tasks Tracker. As shown in figure 1, Job Tracker performs multiple operations which is bottleneck as we keep increasing the number of Task Trackers. Maximum 4000 Task Tracker and 40,000 concurrent tasks, can be handled by single Job Tracker. This was causing the limitation on use of number of Tasks Trackers. In MapReduce, there was a hard partition on resource utilization, which was causing inefficient use of resources [5].

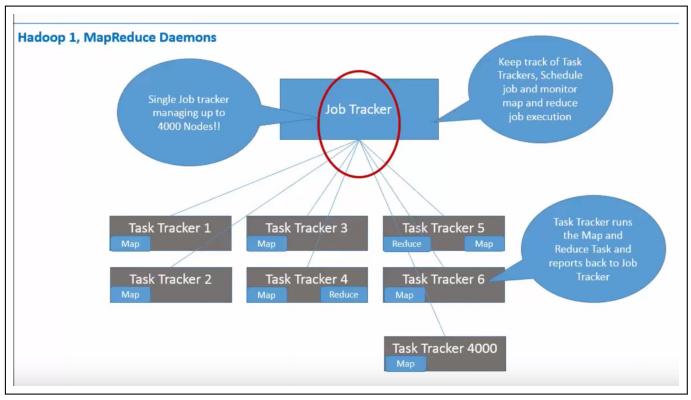


Fig. 1. Hadoop 1, MapReduce deamon diagram [5].

3. DESIGN

To overcome the limitation of MapReduce, Hadoop 2.0 was introduced with Yarn. Yarn is an architectural center of Hadoop 2.0 design [6]. It allows multiple data processing engines like Batch Streaming [5], Interactive SQL [5], Data science and Real time streaming to store and retrieve data in Hadoop Distributed File System (HDFS). Figure 2 showing the Hadoop 2.0 architectural design. Yarn is pre-requisite to Enterprise Hadoop. It acts as middle layer in Hadoop cluster which extends the Hadoop capability which provides resource management, delivers consistent operation, security and data governance tools across Hadoop cluster. It provides ISV engines with a consistent framework which allows developer to write data access applications that run in Hadoop. ISV's (in a context of Hadoop Yarn) create the software product to support and run on the Hadoop Yarn architecture.

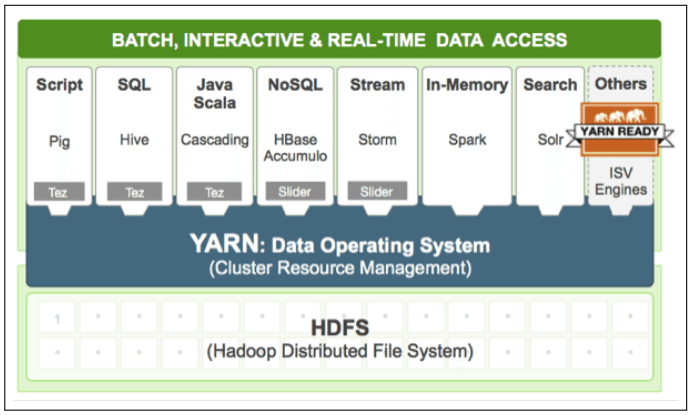


Fig. 2. Architecture diagram for Hadoop 2.0 Yarn[6].

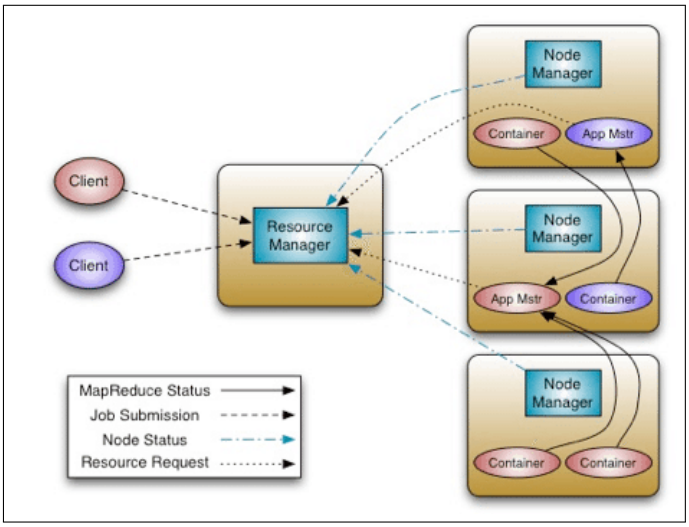


Fig. 3. Hadoop 2.0 Yarn architectural workflow diagram [2].

4. ARCHITECTURE

Apache Hadoop YARN architecture has ResourceManager, ApplicationManager, NodeManager and distributed application as its important parts. Figure 3 shows the architectural work flow of Yarn. YARN's global ResourceManager runs in a master daemon. Usually ResourceManager is placed in dedicated machine, it allocates the requested resource based on available cluster resources. It keeps track of how many live nodes and resources are available on cluster for allocation. It coordinates with the NodeManager for resource allocation and release functionality. ResourceManager consists of two major components Scheduler and ApplicationManager [7]. Scheduler is pure scheduler which only allocates the resource, it will not perform any monitoring or tracking of status. Scheduler schedules the resource (like cpu, disk, network etc.) based on applications requirement. Scheduler can divide the cluster resource among the application tasks, queues it is called as pluggable policy [7]. Once user submits the application to the ResourceManager, the ResourceManager's scheduler will allocate the resource to it. Once resource get allocated ApplicationManger comes into picture and perform the coordination all the tasks running for that application like: monitoring of tasks, restarting failed tasks, speculatively running slow tasks and total values of application counter. It also asks for appropriate resources to run tasks [6]. This work flow is shown in figure 3.

NodeManager will create the container to perform the given tasks. Container size depends on the tasks he needs to perform. Container can be any resource type like CPU, disk, network, and storage. NodeManger can have number of containers depending on the configuration parameter and node resource capacity. Each application has its ApplicationManager. It performs required tasks inside a container. E.g. the MapReduce. ApplicationMaster performs the map and reduce tasks. On similar line Giraph's ApplicationMaster perform Giraph specific tasks in a container. The ResourceManger, the NodeManager and a container work regardless of the type of application. This Yarn feature, makes it more popular, as this allows to run different types of application in a single cluster. This generic approach allows Hadoop Yarn cluster to run various application like MapReduce, Giraph, Storm, Spark, Tez/Impala, MPI etc. To understand the popularity of the Hadoop Yarn, we need to know its advantages.

4.1. Advantages

- Resources can be shared among different clusters, which maximizes the efficiency of resource utilization.
- All in one cluster will help to run maximum application, will reduce the operation cost.
- Reduce the data transfer operation, as there is no need to transfer the data between Hadoop Yarn and system running on different clusters of machine [6].

5. FEATURE

Yarn came with lots of features; we are not covering each of the features here. However, this article will be covering the most important feature, compared to MapReduce, which makes Yarn popular [6].

- **Uberization:** To reduce the overhead on ResourceManager, a small tasks container can ask NodeManager to start their tasks. So all the small tasks of MapReduce are run by ApplicationMaster's JVM. This improves the performance.
- **Multi-tenancy:** Yarn allows multiple engines to access the Hadoop as a common platform for batch processing, interactive and real time data access. This feature returns the most enterprise investment for business.
- **Cluster Utilization:** This feature overcomes the limitation of not using hardware resource efficiently for jobs in MapReduce. Yarn dynamically allocates the resource which improves the resource utilization.
- **Scalability:** Introduces cluster manager so ResourceManager can solely focus on scheduling and let cluster to track the live nodes and available resources in the cluster and assign the tasks to them.
- **Computability:** Existing MapReduce application can run on the latest Hadoop2 Yarn, without failing.

6. LIMITATION

Any new offering takes time to mature and align with market expectation. There are following limitations of Yarn. As the product is growing it is coming with better ecosystem to overcome its limitation [8]:

- Complexity

  - Yarn is complex and level of abstraction is low, from the perspective of developers.
  - Developers need to do very low level tasks, for example: a Hello World program is of 1500 lines code.
  - Clients need to be prepared with all the dependencies (it is mostly library files which help the current program to run with additional functionality).
  - To up and run the app, client requires to setup the environment, like class path.
  - To see the log, developer needs to wait till completion of the job, currently there is no mechanism to show the console log while the job is running.

- Application cannot handle the master crash, which causes the single point failure of the app
- There is no *built in* communication layer available between master and container.
- Not helpful in long running Jobs
- Hard to debug

7. CONCLUSION

Apache Hadoop Yarn is important part of Hadoop 2.0 architecture, which separates the resource management and processing components. With this functionality it achieves scalability, efficiency and flexibility compared to MapReduce engine (from first version of Hadoop). Adding to this Yarn based architecture will not limit the existing MapReduce applications. It supports such existing application to run in Hadoop 2.0, without failure. Most of the Hadoop 1.0 users are migrating to Hadoop 2.0. Yarn runs the large scale data, Yahoo deployed 35,000 nodes for 6 months without failure. Yarn allows multiple access engines like Pig, Hive, JavaScala, NoSQL, Spark and also engines from independent vendor to use the HDFS using Hadoop Yarn. This increased the Hadoop Yarn compatibility to larger extend. YAHOO!, eBay, Spotify, Xing, Allegro and more are already using the Hadoop Yarn.

ACKNOWLEDGEMENTS

Special thanks to Professor Gregor Von Laszewski for giving me an opportunity to write this paper and the most valuable suggestions. Also the entire class and TA's for their suggestion and support.

REFERENCES

[1] Apache Hadoop Developer, "Hadoop Documentation," Web Page, 2017, page last accessed on 12 April 2017. [Online]. Available: http://hadoop.apache.org/

[2] Arun Murthy, "Apache Hadoop Yarn Background An Overview," Web Page, 2012, page last accessed on 7 April 2017. [Online]. Available: https://hortonworks.com/blog/apache-hadoop-yarn-background-and-an-overview/

[3] Apache Hadoop Developers, "Apache Hadoop Release," Web Page, 2017, page last accessed on 12 April 2017. [Online]. Available: http://hadoop.apache.org/releases.html

[4] MapReduce Hadoop Developers, "MapReduce Tutorial," Web Page, 2013, page last accessed on 12 April 2017. [Online]. Available: https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html

[5] Kawa Adam, "Introduction to YARN," Web Page, 2014, page last accessed on 7 April 2017. [Online]. Available: https://www.ibm.com/developerworks/library/bd-yarn-intro/

[6] Horton works, "APACHE HADOOP YARN," Web Page, page last accessed on 7 April 2017. [Online]. Available: https://hortonworks.com/apache/yarn/

[7] The Apache Software Foundation, "Apache Hadoop Yarn," Web Page, 2016, page last accessed on 7 April 2017. [Online]. Available: https://hadoop.apache.org/docs/r2.7.2/hadoop-yarn/hadoop-yarn-site/YARN.html

[8] BigDSol, "Understand Hadoop YARN," YouTube, 2013, page last accessed on 7 April 2017. [Online]. Available: https://www.youtube.com/watch?v=1vg_W-MMZpA

Apache Tez- Application Data processing Framework

ABHIJT THAKRE[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[2] *Mechanical Engineer,Nagpur University, 2003*
[*] *Corresponding authors: abhijit.thakre@gmail.com*

project-000, April 30, 2017

There are lot of advancement in Hadoop framework from Hadoop1.0 to Hadoop 2.0. Hadoop 2.0 is layered architecture provides the YARN layer responsible for the resource management opens up the gate for developing different application engines on top of it. Apache Tez is one of such open source framework build on the top of YARN designed to build data-flow driven runtimes. This paper focusses mostly on introduction to Apache Tez framework. It provides the insight of the architecture used in building Tez. It also tried to cover the technologies using apache Tez as unifying framework and performance improvement achieved due to that.

Keywords: Cloud, I524

https://github.com/cloudmesh/classes/blob/master/docs/source/format/report/report.pdf

1. INTRODUCTION

In order to understand Tez as an framework we need to first dig into the history of Hadoop. Hadoop 1.0 have MapReduce as the central execution engine of its application. Any type of problem statement for analysis needs to be restructured to fit it to the map-reduce paradigm. It was also responsible for resource management and resource allocation. With Hadoop 2.0 these responsibilities got divided separately where YARN got the responsibility of general purpose resource management.

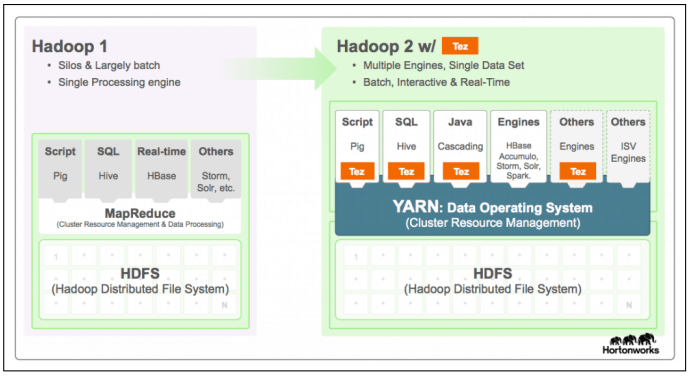


Fig. 1. [1]

Apache Hive, Apache PIG , CASCADING which was initially using Hadoop1 now run on Hadoop2 on YarN. These listed data processing application including map reduce have certain set of requirement from the hadoop cluster in order to run efficiently. This is where Tez came in picture. Tez takes care of running the data processing application's efficiency and performance leaving the end user to only concentrate on the business logic.

2. TEZ TERMINOLOGY

DAG – Direct Acyclic Graphs, it represent overall job. Vertex – Logical step in processing. It contains the details of user logic and dependent environment. Task – There can be multiple task in unit of work that vertex perform. Edge – This represents connection between producer and consumer vertices.

3. ARCHITECTURE AND IMPLEMENTATION

Tez was designed keeping in mind to address the problems which were not resolved by Hadoop. It was not build from the scratch but on the top of YARN layer to leavarage the advantages and work that were done for years on Hadoop. So Tez leavarage the discrete task based compute model,concept of data shuffling in map reduce,resource tentancy and multi-tenancy model and build in security from Hadoop .

Tez focuses mainly on below problem in addition.

- Without Tez, all the algorithm those needs to be executed on clusters needs to somehow translated to map-reduce api. This was impacting the efficiency and performance.However with Tez can naturally map the algorithm to execution engine in cluster.
- Tez provide additionally interface for various application and technology for data source and syncing.
- Performance.

4. TEZ-API

Tez provides below two API for defining the data processing.

4.1. DAG API

This API lets user define the structure for the computation. It lets user define the producer and consumers and how they talk to each other. This class of data processing application is represented as direct acyclic graphs

4.2. RUNTIME API

Using this API Layer Tez invokes the user code. This is where the actual code in the task to be executed is defined.

5. APPLICATION USING TEZ

Below are the product those are updated to be used on TEZ framework to run on YARN.

5.1. Apache MapReduce

MapReduce is simple but powerful way of data processing. Tez product comes with build in map-reduce support.The configuration needs to be updated on YARN cluster for map reduce. Tez has inbuild map processor and reduce processor which provides the respective map reduce functionality.

5.2. Apache PIG

PigLatin is the scripting language provided by Apache PIG used to write complex ETL. Tez API can handle the multiple output which is usually the case for the procedural language like PIGlatin, helps in keeping the code clean and maintainable.

5.3. Apache Hive

Apache Hive is used to convert the query written in HiveQL to map reduce and execute on hadooop cluster.The HiveQL is translation into map reduce format is often inefficient. Hive 0.13 was developed using Tez integration [2] and the trees translate directly into DAGS.Hive0.14 have additional improvements like dynamic partition prunning.

5.4. Apache Spark

Aparche spark provides scaLA API for distributed data processing. The output of the spark is DAG of tasks that performance distributed computation. The end output of Apache spark i.e Spark DAG is successfully converted and executed using Tez API.

6. COMPLEX HIVE/PIG JOB RUNNING ON MR VS TEZ

API library, YARN application master and runtime library are three major pillar used by apache Tez project.Tez application DAG represents the flow of the application with input and output as key-value pair for ease of use.

Tez leavarage all the best ways used in distributed data processing application for improving the efficient and performance. To quote few mechanism Tez uses like running the task close to its data,monitoring of slaggered task and running them in parallel to take them to finish, reusing the contain and sessions.

The following figure depicts, earlier running complex script and queries on Map Reduce use to take multiple jobs. Running those job on Tez have reduced the number of steps.

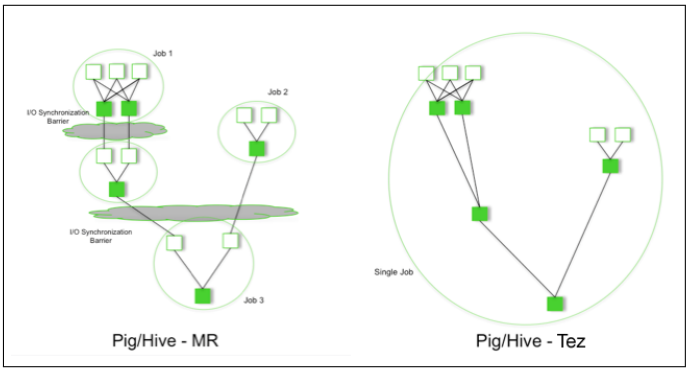


Fig. 2. [1]

7. PERFORMANCE RESULTS

Various test have been conducted to measure the performance of Hive and Pig implemented on the top of Tez vs its original implementation on the Yarn. The below section details the configuration and test results for both of them.

7.1. HIVE

TEZ provides features like broadcast edges, runtime reconfiguration and custom vertex manager which improves the overall performance of the Hive. The figure below depicts the comparison for Hive running on TEZ vs old map reduce implementation for the work load of 30 tb scale on 20 nodes clusters on 16 cores. Each node was assigned 256 Gb RAM and 6* 4 TB drives.

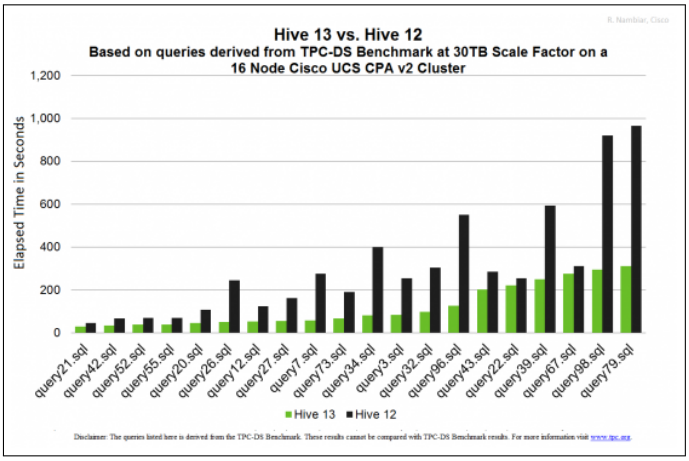


Fig. 3. [1]

One of the test results published in San Jose Hadoop Summit shows that Hive with Tez outperforms for TPC-H workload as compared to Hive with Map Reduce [3].

7.2. PIG

Yahoo have conducted production for complex ETL job. The load used for testing encompass 1.100K Task 2.Input in TBs 3.Complex DAGs. 4.Combination of complex queries.

The configuration of environment/clusters used for the test 1.Cluster has 4200 server 2.46PB hdfs storage 3.90TB aggregate memory 4.24GB RAM 5.2x Xeon 2.40GHz , 6*2TB SATA on Hadoop 2.5, RHEL 6.5

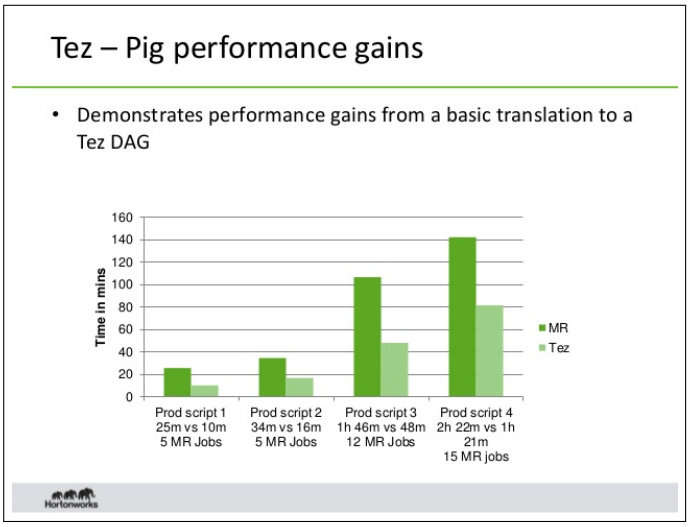


Fig. 4. [1]

The performance improvement 150 to 200% is observed with TEZ.

7.3. Spark Multi-Tenancy Test

Tez allocates the resources based on task as compared to YARN which allocated based on the job life cycle. Spark on TEZ implementation have shown considerable speed up as the idle resource in TEZ based implementation get allocated to next job as compared to the service based where the resource wait till the life cycle end . This has been proved based on the 4- user concurrency test of partitioning a TPC-H lineitem data set. The allocation graph is shown in figure 5 and 6.

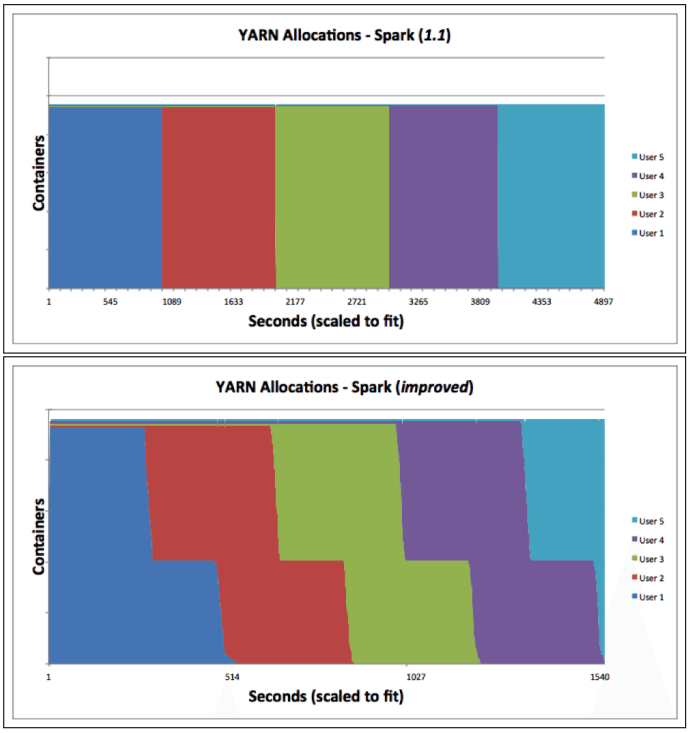


Fig. 5. [4]

8. CONCLUSION

With growing trend of big data and its processing, Hadoop has been in demand. Tez is open source architecture build modeling the data processing as direct acyclic graphs.

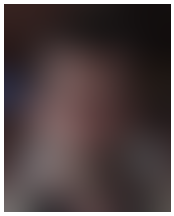
It has leverages all the strong points from Hadoop and addressed the issues which were concerning to the performance.

Various examples discussed in paper shows how the implementation of various systems like HIVE,PIG, SPARK on TEZ have shown substantial performance improvement as compared to the traditional MR api. Tez is open source and high customizable framework. It allows researchers and open source developer to integrate test their code.

REFERENCES

[1] "TEZ," Web Page, Indiana University, Mar. 2017. [Online]. Available: http://www.tez.apache.org
[2] "TEZ," Web Page, Indiana University, Apr. 2017. [Online]. Available: http://issues.apache.org/jira/browse/HIVE-4660
[3] "TEZ," Web Page, Indiana University, Apr. 2017. [Online]. Available: http://yahoodevelopers.tumblr.com/post/85930551108/yahoo-betting-on-apache-hive-tez-and-yarn.
[4] "TEZ," Web Page, Indiana University, Apr. 2017. [Online]. Available: http://sungsoo.github.io/2015/06/06/spark-on-yarn.html

AUTHOR BIOGRAPHIES



Abhijit Thakre received his BE (Mechanical) in 2003 from The University of Nagpur.

Deployment Model of Juju

SUNANDA UNNI[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: suunni@indiana.edu

paper2, April 30, 2017

Developing an application for the cloud is accomplished by relying on the Infrastructure as a Service (IaaS) or the Platform as a Service (PaaS). Juju is a software from Canonical that provides open source service orchestration using a model of IaaS. Juju charms can be deployed for IaaS on cloud services such as Amazon Web Services (AWS), Microsoft Azure and OpenStack. Deep Server provisioning is provided by Juju using MAAS, Metal as a Service. We are exploring the deployment model of Juju.

Keywords: Cloud, I524, Juju, IaaS

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IO-3022/report.pdf

1. INTRODUCTION

Deploying software component systems [1] is becoming a critical challenge, especially due to the advent of Cloud Computing technologies that make it possible to quickly run complex distributed software systems on-demand on a virtualized infrastructure at a fraction of the cost compared to a few years ago. When the number of software components needed to run the application grows, and their interdependencies become too complex to be manually managed, it is important for the system administrator to use high-level languages for specifying the expected minimal system.

2. IAAS, PAAS AND MAAS

The IaaS provides a set of low-level resources forming a bare computing environment. Developers pack the whole software stack into virtual machines containing the application and its dependencies and run them on physical machines of the provider's cloud. Exploiting the IaaS directly allows a great flexibility but requires also a great expertise and knowledge of the cloud and application entities involved in the process IaaS describes the provision of processing, storage and networking (and potentially) other basic computing resources, over a network and in an on-demand fashion. An example of IaaS is the AWS[2], Juju.

IaaS customers does not host or manage the dedicated or virtual server. Figure 1 for Juju scheme of work as IaaS.

In PaaS (e.g., Google App Engine [3], Azure [4]) a full development environment is provided. Applications are directly written in a programming language supported by the framework offered by the provider, and then automatically deployed to the cloud. The high-level of automation comes however at the price of flexibility: the choice of the programming language to use is restricted to the ones supported by the PaaS provider,

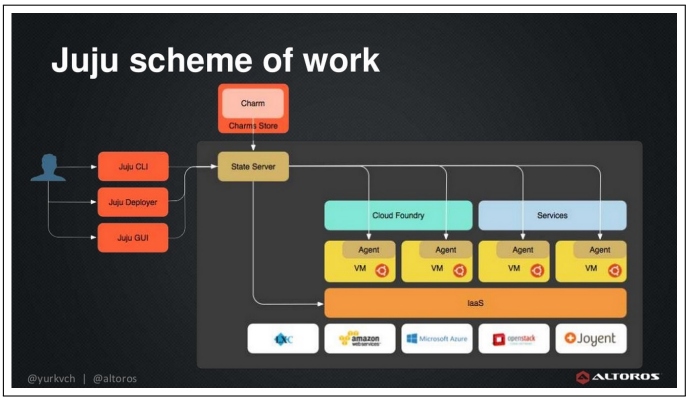


Fig. 1. Basic Deployment of Juju scheme of work.

and the application code must conform to specific APIs.

3. JUJU

In the IaaS, two deployment approaches [1] are gaining more and more momentum: the holistic and the DevOps one. In the former, also known as model-driven approach, one derives a complete model for the entire application and the deployment plan is then derived in a top-down manner. In the latter, put forward by the DevOps community [5], an application is deployed by assembling available components that serve as the basic building blocks. This emerging approach works in a bottom-up direction: from individual component descriptions and recipes for installing them, an application is built as a composition of these recipes.

One of the representative for the DevOps approach is Juju [6], by Canonical. It is based on the concept of charm: the atomic

Fig. 2. Juju installation steps[6]

```
$ sudo apt-get update && sudo apt-get install juju
```

Fig. 3. Juju bootstrap command with yaml configuration set [6].

```
$ sudo juju generate-config
$ edit the environments.yaml file
$ sudo juju sync-tools
$ sudo juju bootstrap
```

unit containing a description of a component.

In 2.0, Juju follows a Controller-Model type of configuration. A Juju controller is the management node of a Juju cloud environment. In particluar, it houses the database and keeps track of all the models in that environment. Although it is a special node, it is a machine that gets created by Juju (during the "bootstrap" stage) and, in that sense, is similar to other Juju machines.

A Juju model is an environment associated with a controller. During controller creation two models are also created, the 'controller' model and the 'default' model. The primary purpose of the 'controller' model is to run and manage the Juju API server and the underlying database. Additional models may be created by the user.

Since a controller can host multiple models, the destruction of a controller must be done with ample consideration since all its models will be destroyed along with it.

4. JUJU INSTALLATION

Installing Juju is straight forward, it is described in detail in Getting Started document [7] and Paper regarding deploying Juju on MaaS [8].

The command to install Juju is shown in Fig 2.

Not covering setup of LXD or MaaS for the physical containers or cluster on which Juju is to be installed. We can refer to [7] for LXD and [8] for MaaS installation.

To create a controller, juju bootstrap command is used. Two ways to bootstrap

1. Generate config with the environment specified in yaml, as shown in Fig 3.

2. By specifying the details on command line as shown in Fig 4.

5. INSTALLATION OF CHARMS

Fig 5 shows the deployment model of Charms.

When deploying charms you can do the following-

Fig. 4. Juju bootstrap command with command line settings [6].

```
$ sudo juju bootstrap localhost lxd-test

lxd-test is the controller on the local machine.
```

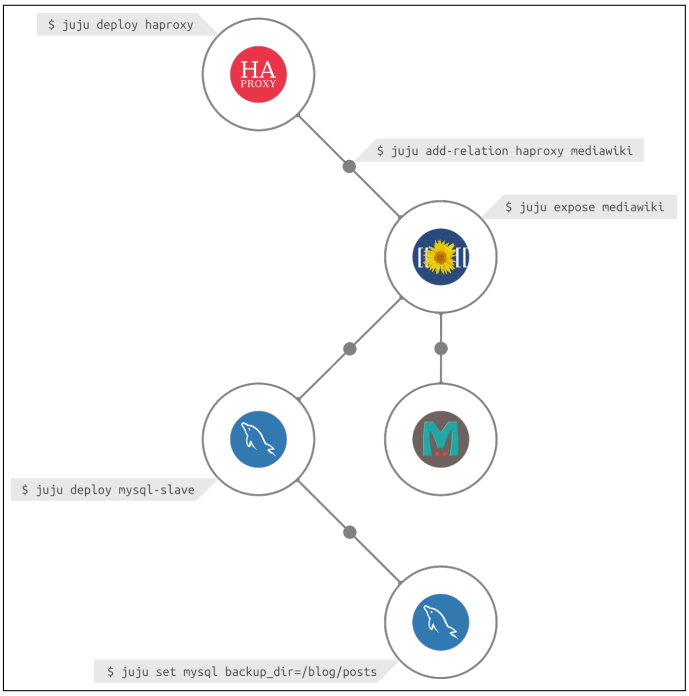


Fig. 5. Charms and Bundles [6].

Fig. 6. Juju Charm deploy from external repository [6].

```
$ sudo apt-get install charm-tools
$ charm get <charm>
$ juju deploy local:trusty/<charm-name>
```

1. Download them from external repositories as you deploy. For direct installation from external repository, we can use Fig 6.

2. Download them first to a local repository and then point to the repository as you deploy. For deploying from local repository you can install bazar as shown in Fig 7.

Status of installation can be checked by command as shown in Fig 8.

On successful installation of MySQL and mediawiki, we can see the status as shown in Fig 9.

6. BUNDLING

A Bundle is an encapsulation of complex deployments including many different applications and connections. Bundles are deployed as-

1. From the Juju Charm Store as shown in Fig 10.

Fig. 7. Juju Charm deploy from local repository [8].

```
$ sudo apt-get install bzr
$ mkdir -p /opt/charms/trusty; cd /opt/charms/trusty
$ bzr branch lp:charms/<charm-name>
$ juju deploy repository=/opt/charms local:trusty/<charm-nar
```

Fig. 8. Juju Charm installation status [6].

```
$ juju status
```

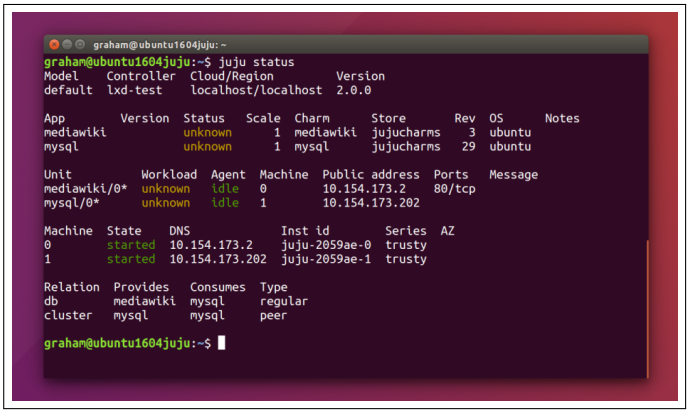


Fig. 9. Juju status post installation of Bundle [7].

2. By exporting the current configuration into a bundle. Juju-UI can be used for exporting the current configuration

7. ADVANTAGES OF JUJU

Scaling the setup is possible. No prior knowledge of application stack required. Charm store provides charms for a lot of common Opensource applications. Bundling helps in easy re-deployment to new environment. Juju works with the exiting Configuration management tool.

8. DRAWBACKS OF JUJU

In order to use Juju, some advanced knowledge of the application to install is mandatory. This is due to the fact that the metadata does not specify the required functionalities needed by a component. Currently Juju can be used for deployments on limited OS - Windows, CentOS, and support for Ubuntu. Juju does not still support handling of circular dependencies

9. CONCLUSION

For Application provisioning Juju does provides orchestration similar to Puppet, Chef, Salt etc. Advantage is Juju comes with GUI in open source version and is interoperable with most of the major cloud providers.

REFERENCES

[1] T. A. Lascu, J. Mauro, and G. Zavattaro, "Automatic deployment of component-based applications," *Science of Computer Programming*, vol. 113, pp. 261–284, 2015.

[2] Amazon Web Services, Inc., "Amazon web services," Web Page, 2017. [Online]. Available: https://aws.amazon.com/

[3] Google Developers, "Google app engine documentation," Web Page. [Online]. Available: https://developers.google.com/appengine

[4] Microsoft, "Microsoft azure," Web Page, 2017. [Online]. Available: http://azure.microsoft.com

[5] Mediaops, LLC, "Devops- where the world meets devops," Web Page, 2017. [Online]. Available: https://devops.com

[6] Canonical Ltd, "Ubuntu cloud documentation," Web Page. [Online]. Available: https://www.ubuntu.com/cloud/juju

[7] Canonical Ltd, "Getting started with juju," Web Page. [Online]. Available: https://jujucharms.com/docs/stable/getting-started

[8] K. Baxley, J. la Rosa, and M. Wenning, "Deploying workloads with juju and maas in ubuntu 14.04 lts," Dell Inc. Dell Inc, may 2014. [Online]. Available: http://en.community.dell.com/techcenter/extras/m/white_papers/20439303

Fig. 10. Deploy bundle from Juju Charm Store [7].

```
juju deploy cs:bundle/wiki-simple-0
```

AWS Lambda

KARTHICK VENKATESAN[1,*,+]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: vkarthickprabu@gmail.com
[+] HID - S17-IO-3023

paper2, April 30, 2017

The rapid pace of innovation in data centers and the software platforms within them is set to transform how we build, deploy, and manage online applications and services. Common to both hardware-based and container-based. Virtualization is the central notion of a server. Servers have been used for the past several years to back online applications, but new cloud-computing platforms foreshadow the end of the traditional backend server. Servers are notoriously difficult to configure and manage, and server startup time severely limits an application's ability to scale up and down quickly. As a result, a new model, called serverless computation, is poised to transform the construction of modern, scalable applications ability to quickly scale up and down . This paper is on AWS Lambda a Serverless Computing technology.

Keywords: AWS Lambda,Serverless Computing, I524

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IO-3023/report.pdf

1. INTRODUCTION

Serverless applications are where some amount of server-side logic, unlike traditional architectures, is run in stateless compute containers that are event-triggered, ephemeral, and fully managed by a 3rd party. One way to think of this is 'Functions as a service / Faas'. AWS Lambda is one of the most popular implementations of Faas [1].

AWS Lambda is a FaaS(Function as a Service) from Amazon Web Services. It runs the backend code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance and capacity provisioning [2].

In AWS Lambda one needs to pay only for the compute time consumed - there is no charge when the code is not running. AWS Lambda, can run code for virtually any type of application or backend service with zero administration. AWS Lambda requires only the code to be uploaded, and Lambda takes care of everything required to run and scale the code with high availability. AWS Lambda can be setup to automatically trigger the code from other AWS services or call it directly from any web or mobile app [3].

2. FEATURES

The key features of AWS Lambda are [3]

- **No Servers to Manage**: AWS Lambda automatically runs the code without requiring to provision or manage servers. Just write the code and upload it to Lambda.
- **Continuous Scaling**: AWS Lambda automatically scales the application by running code in response to each trigger. The code runs in parallel and processes each trigger individually, scaling precisely with the size of the workload.
- **Subsecond Metering**: With AWS Lambda, the institution using the service are charged for every 100ms the code executes and the number of times the code is triggered. They don't pay anything when the code isn't running.

3. ARCHITECTURE

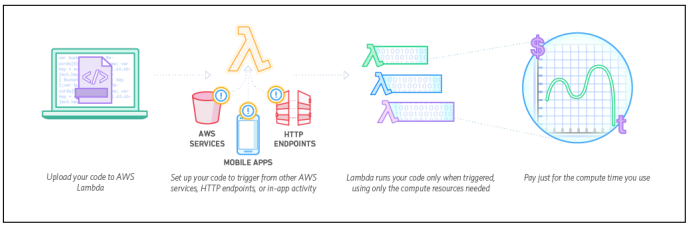


Fig. 1. AWS Lambda Architecture [3]

Lambda functions and event sources are the core components in AWS Lambda. An event source is the entity that publishes events, and a Lambda function is the custom code that processes the events. Several AWS cloud services can be preconfigured to work with AWS Lambda. The configuration is referred to as event source mapping, which maps an event source to a Lambda

function. It enables automatic invocation of Lambda function when events occur.

Each event source mapping identifies the type of events to publish and the Lambda function to invoke when events occur. The specific Lambda function then receives the event information as a parameter, and as shown in Figure 1 the Lambda function code can then process the event.

The event sources can be any of the following:

AWS services – These are the supported AWS services that can be preconfigured to work with AWS Lambda. These services can be grouped as regular AWS services or stream-based services. Amazon Kinesis Streams [4] and Amazon DynamoDB Streams [5] are stream-based event sources, all others AWS services do not use stream-based event sources.

Custom applications – Custom applications built can also publish events and invoke a Lambda function.

4. SCALABILITY

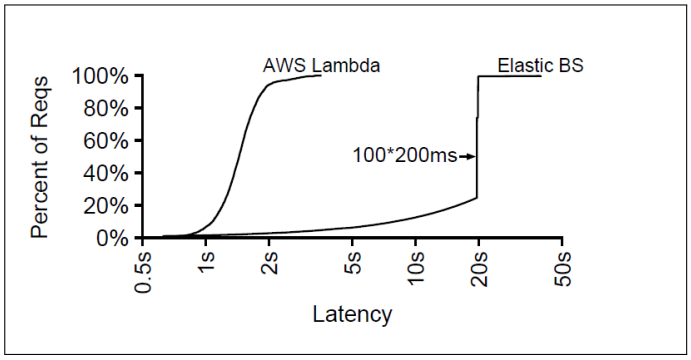


Fig. 2. Response Time. This CDF shows measured response times from a simulated load burst to an Elastic BS application and to an AWS Lambda application.
[6]

A primary advantage of the Lambda model is its ability to quickly and automatically scale the number of workers when load suddenly increases. The Graph in Figure 2 demonstrates this by comparing AWS Lambda to a container-based server platform, AWS Elastic Beanstalk [7] (hereafter Elastic BS). On both platforms, the same benchmark was run for one minute: the workload maintains 100 outstanding RPC(Remote Procedure Calls) requests and each RPC handler spins for 200ms. Figure 2 shows the result: an RPC using AWS Lambda has a median response time of only 1.6s, whereas an RPC in Elastic BS often takes 20s. Investigating the cause for this difference, it was found that while AWS Lambda was able to start 100 unique worker instances within 1.6s to serve the requests; all Elastic BS requests were served by the same instance; as a result, each request in Elastic BS had to wait behind 99 other 200ms requests. AWS Lambda also has the advantage of not requiring configuration for scaling. In contrast, Elastic BS con- figuration is complex, involving 20 different settings for scaling alone. Even though the Elastic BS was tuned to scale as fast as possible, it still failed to spin up new workers for several minutes [6].

5. DOCUMENTATION

- Detailed documentation on AWS Lambda Deployment , Configuration,Debugging and Development is available at [8].
- Use cases on reference architecture with AWS Lambda are available at [9].

6. COMPETITORS

The main competitors for AWS Lambda are

- Microsoft Azure Functions
- Google Cloud

A detailed comparison of features between AWS Lambda, Google Cloud and Microsoft Azure Functions is available in Table 1.

7. PRICING

With AWS Lambda, the users pay only for what they use. They are charged based on the number of requests for the functions and the duration the code executes.

Requests: The users are charged for the total number of requests across all their functions. Lambda counts a request each time it starts executing in response to an event notification or invokes call, including test invokes from the console. First 1 million requests per month are free $0.20 per 1 million requests thereafter ($0.0000002 per request)

Duration:Duration is calculated from the time the code begins executing until it returns or otherwise terminates, rounded up to the nearest 100ms. The price depends on the amount of memory the user allocates to the function. The users are charged $0.00001667 for every GB-second used [11].

8. USE CASE

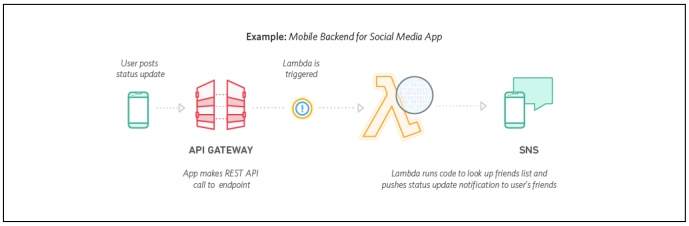


Fig. 3. Mobile Backend
[3]

Mobile Backends: Mobile Backends are an excellent use case to implement AWS Lambda.As shown in Figure 3 the back-end of the mobile application can be built using AWS Lambda and Amazon API Gateway to authenticate and process API requests. Lambda makes it easy to create rich, personalised app experiences. **Bustle.com** is a news, entertainment, lifestyle, and fashion website catering to women. Bustle also operates **Romper.com**, a website focused on motherhood.Bustle is based in Brooklyn, NY and is read by 50 million people each month. Bustle uses AWS Lambda to process high volumes of site metric data from Amazon Kinesis Streams [4] in real time. This allows the Bustle team to get data more quickly so they can understand how new site features affect usage. They can also measure user engagement, allowing better data-driven decisions. The serverless back end supports the Romper website and iOS app as well as the Bustle iOS app. With AWS Lambda, Bustle was able to eliminate the need to worry about operations. The engineering team at Bustle focus on just writing code, deploy it, and it

Table 1. AWS Lambda vs. Google Cloud Functions vs. Microsoft Azure Functions [10]

FEATURE	AWS LAMBDA	GOOGLE CLOUD	AZURE FUNCTIONS
Scalability & availability	Automatic scaling (transparently)	Automatic scaling	Metered scaling (App Service Plan) Automatic scaling (Consumption Plan)
Max # of functions	Unlimited functions	20 functions per project (alpha)	Unlimited functions
Concurrent executions	100 parallel executions (soft limit)	No limit	No limit
Max execution	300 sec (5 min)	No limit	300 sec (5 min)
Supported languages	JavaScript, Java Python, C#	Only JavaScript	C# and JavaScript (preview of F#, Python, Batch, PHP, PowerShell)
Dependencies	Deployment Packages	npm package.json	Npm, NuGet
Deployments	Only ZIP upload (to Lambda or S3)	ZIP upload, Cloud Storage or Cloud Source Repositories	Visual Studio Team Services, OneDrive, GitHub, Bitbucket, Dropbox
Environment variables	Yes	Not yet	App Settings and ConnectionStrings from App Services
Versioning	Versions and aliases	Cloud Source branch/tag	Cloud Source branch/tag
Event-driven	S3, SNS, SES, DynamoDB, Kinesis, CloudWatch, Cognito, API Gateway	Cloud Pub/Sub or Cloud Storage Object Change Notifications	Blob, EventHub, Generic WebHook Queue, Http, Timer triggers
HTTP(S) invocation	API Gateway	HTTP trigger	HTTP trigger
Orchestration	AWS Step Functions	Not yet	Azure Logic Apps
Logs management	CloudWatch	Cloud Logging	App Services monitoring
In-browser code editor	Yes	Only with Cloud Source Repositories	Functions environment, AppServices editor
Granular IAM	IAM roles	Not yet	IAM roles
Pricing	1M requests for free (Free Tier), then $0.20/1M requests	Unknown until open beta	1 million requests and 400,000 GB-s

scales infinitely, and they don't have to deal with infrastructure management. Bustle was able to achieve the same level of operational scale with half the size of team of what is normally needed to build and operate the site [12].

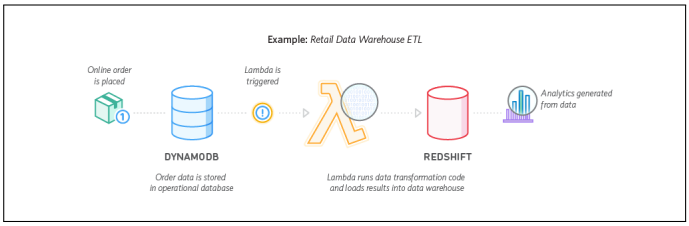


Fig. 4. Lambda ETL [3]

Extract, Transform, Load: As shown in figure 4 AWS Lambda can be used to perform data validation, filtering, sorting, or other transformations for every data change in a DynamoDB table and load the transformed data to another data store. **Zillow** is the leading real estate and rental marketplace dedicated to empowering consumers with data, inspiration and knowledge around the place they call home, and connecting them with the best local professionals who can help. Zillow needed to collect a subset of mobile app metrics in realtime and report it to the buisness users several times during the day.The solution was to be delivered in 3 weeks.Leveraging AWS Lambda and Amazon Kinesis Zillow was able to seamlessly scale 56 lines of code to over 16 million posts a day and achieve its goal in 2 weeks [13].

9. CONCLUSION

Serverless Computing allows to build and run applications and services without thinking about servers. At the core of serverless computing is AWS Lambda, which lets to build auto-scaling, pay-per-execution, event-driven apps quickly.

ACKNOWLEDGEMENTS

The authors thank Prof. Gregor von Laszewski for his technical guidance.

REFERENCES

[1] M. Roberts, "Serverless," Web Page, Aug. 2016, accessed 2017-03-26. [Online]. Available: https://martinfowler.com/articles/serverless.html

[2] A. Gupta, "Serverless faas with aws lambda and java," Web Page, Jan. 2017, accessed 2017-03-26. [Online]. Available: https://blog.couchbase.com/serverless-faas-aws-lambda-java/

[3] Amazon Web Services, Inc, "AWS Lambda," Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/lambda/

[4] Amazon Web Services, Inc, “AWS Kinesis Streams,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/kinesis/streams/

[5] Amazon Web Services, Inc, “AWS Dynamo DB Streams,” Web Page, accessed 2017-03-26. [Online]. Available: http://docs.aws.amazon.com/amazondynamodb/latest/APIReference/API_Types_Amazon_DynamoDB_Streams.html

[6] S. Hendrickson, S. Sturdevant, T. Harter, V. Venkataramani, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau, “Serverless computation with openlambda,” in *8th USENIX Workshop on Hot Topics in Cloud Computing, HotCloud 2016, Denver, CO, USA, June 20-21, 2016*. Berkeley, CA, USA: USENIX Association, 2016. [Online]. Available: https://www.usenix.org/conference/hotcloud16/workshop-program/presentation/hendrickson

[7] Amazon Web Services, Inc, “AWS Elastic Beanstalk,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/elasticbeanstalk/

[8] Amazon Web Services, Inc, “AWS Lambda Doc,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/documentation/lambda/

[9] Amazon Web Services, Inc, “AWS Lambda Use Case,” Web Page, accessed 2017-03-26. [Online]. Available: http://docs.aws.amazon.com/lambda/latest/dg/use-cases.html

[10] M. Parenzan, “Microsoft azure functions vs. google cloud functions vs. aws lambda,” Web Page, Feb. 2017, accessed 2017-03-26. [Online]. Available: http://cloudacademy.com/blog/microsoft-azure-functions-vs-google-cloud-functions-fight-for-serverless-cloud-domination-continues/

[11] Amazon Web Services, Inc, “AWS Lambda Pricing,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/lambda/pricing/

[12] Amazon Web Services, Inc, “Bustle Case Study,” Web Page, accessed 2017-03-26. [Online]. Available: https://aws.amazon.com/solutions/case-studies/bustle/

[13] Amazon Web Services, Inc, “AWS re:Invent 2015 | (BDT307) Zero Infrastructure, Real-Time Data Collection, and Analytics,” https://www.youtube.com/watch?v=ygHGPnAd0Uo, Youtube, Oct. 2015, accessed 2017-03-26.

LaTeX Puppet - Automatic Configration management

ASHOK VUPPADA[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: ashokmadhu66@gmail.com

project-000, April 30, 2017

Cloud providers today need to deliver pre installed infrastructure , operating system or programing language as per the service level agreements. There would be need to maintain multiple instances with various dependencies installed. It would become extremely impossible to maintain the configurations manually ,hence configuration management tools become a necessary in the cloud era. Puppet is one of the configuration management tool available today
Sharelatex systems.

Keywords: Cloud, I524

https://github.com/justbbusy/sp17-i524/tree/master/paper2/S17-IO-3024/report.pdf

1. INTRODUCTION

Puppet can mean the programming language in which the end state required is defined. unlike the procedural steps there is no need to know the required steps to achieve the end state , puppet would internally manage the required steps and ensures the end state is achieved. The same Puppet language code works between various operating systems. [1]

Puppet Language alone cannot achieve the configuration management required , puppet code needs to be maintained on regular basis to ensure the required dependencies are properly getting installed on the client servers. Puppet platform provides the required framework to maintain the puppet code. [1]

2. ARCHITECTURE

Catalog is the file which contains the end state of required at the client , it is defined in puppet language. This file is maintained at the puppet master, It would be downloaded by puppet client from the puppet master when it is run. The changes are applied by compiling and running the catalog.[2]

Puppet agent is a daemon process runs on the client machine where the configuration is required to be managed. Puppet master is a daemon process that runs on the host which manages the configuration across the various clients. The puppet agent and master would be communicated through a secure SSL connection. The puppet master would keep checking with client if the required installations are done or not , if there is any change in the end state required at a given client the puppet agent would run and ensures the end state is changed as per the configuration. It is also possible to define the time interval required for the puppet master to check with the client.[3]

Puppet is developed by Puppet Labs using ruby language and released as GNU General Public License (GPL) until version 2.7.0 and the Apache License 2.0 after that.[4]

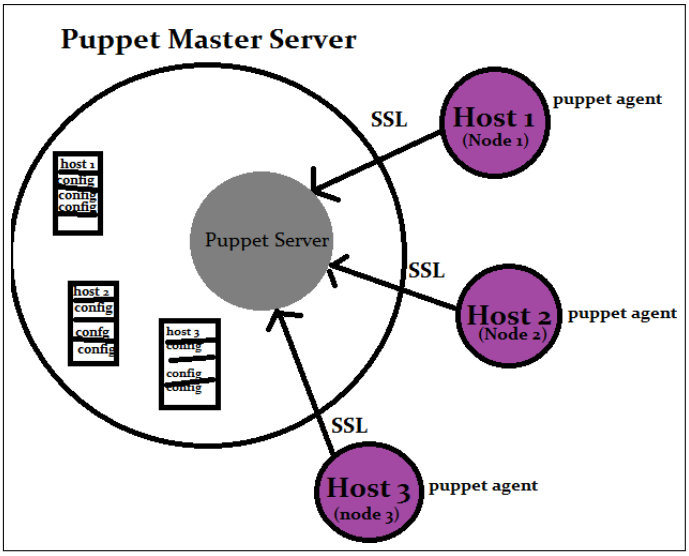


Fig. 1. Puppet Master Puppet Client Communication

3. CONFIGURATION

The below screenshots expains how the configure the puppet master and client and also to make the the puppet client have the changes immediately.

3.1. Puppet Master Configuration

```
#Add Puppet repos
[user@client ~]# sudo rpm -ivh
http://yum.puppetlabs.com/puppetlabs-release-el-6.noarch.rpm

[user@client ~]# sudo yum install puppet

#Open the conf file and add the puppet server hostname
[user@client ~]#sudo vim /etc/puppet/puppet.conf
[main]
# The puppetmaster server
server=puppet.yourserver.com



[user@client ~]# sudo service puppet start
```

Fig. 2. Puppet Master Configuration

[5]

3.2. Puppet Client Configuration

```
#Add Puppet repos
[user@puppet ~]# sudo rpm -ivh
http://yum.puppetlabs.com/puppetlabs-release-el-6.noarch.rpm

[user@puppet ~]# sudo yum install puppet-server

# Add your puppet server hostnames to the conf file under the
[main] section
[user@puppet ~]# sudo vim /etc/puppet/puppet.conf

 dns_alt_names = puppet,puppet.yourserver.com

[user@puppet ~]# sudo  service puppetmaster start
```

Fig. 3. Puppet Client Configuration

[5]

3.3. If Client needs to changes immediately

```
[user@puppet ~]# sudo puppet agent --test
```

Fig. 4. If Client needs to changes immediately

[5]

4. ADVANTAGES

1. Puppet Labs provides very good support for this tool.
2. Good interface and runs on almost all operating systems.
3. Installation and setup is simple
4. Strong reporting capabilities

[6]

5. DISADVANTAGES

1. For more advanced tasks, one need to use the CLI, which is Ruby-based makes it necessary to have ruby knowledge.
2. Support for pure-Ruby versions is being scaled back.
3. DSL and a design that does not focus on simplicity, the Puppet code base can grow large, unwieldy, and hard to pick up for new people in your organization at higher scale.
4. Model-driven approach means less control compared to code-driven approaches.[6]

6. CONCLUSION

The need of configuration is essential with the cloud offering with various tools like Chef , Ansible , Puppet , Salt etc in market gives freedom for the configuration management team to use the the applicable tool based on the case to case need.

REFERENCES

[1] “Introduction to puppet,” web page. [Online]. Available: https://www.infoq.com/articles/introduction-puppet
[2] “Puppet documentation,” web page. [Online]. Available: https://docs.puppet.com/puppet/4.9/architecture.html
[3] “How puppet works,” web page. [Online]. Available: http://www.slashroot.in/puppet-tutorial-how-does-puppet-work
[4] “Puppet software,” web page. [Online]. Available: https://en.wikipedia.org/wiki/Puppet_(software)
[5] “Installing and configuring puppet,” web page. [Online]. Available: https://techarena51.com/index.php/a-simple-way-to-install-and-configure-a-puppet-server-on-linux/
[6] “Puppet vs ansible vs chef,” web page. [Online]. Available: http://www.intigua.com/blog/puppet-vs.-chef-vs.-ansible-vs.-saltstack

HUBzero: A Platform For Scientific Collaboration

NITEESH KUMAR AKURATI[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: akuratin@indiana.edu*

paper2, April 30, 2017

HUBzero is a cyberinfrastructure in a box that is used to create dynamic websites for educational activities as well as scientific research. HUBzero provides an platform where researchers can publish and share their research software and related material for educational purposes on the web. Other researchers will be able to access the tools as well as material using a Web browser and can also launch simulation runs on the national Grid infrastructure without having to download any code.

Keywords: cyberinfrastructure, BigData, Simulation Tools, Collaboration

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2001/report.pdf

1. INTRODUCTION

HUBzero is a platform created to support nanoHUB.org, an online community for the Network for Computational Nanatechnology(NCN), which is funded by the U.S.National Science Foundation since 2002 to connect the theorists across the academia who develop simulation tools with the educators and the experimentalists who might use them. HUBzero has been expanded to more than 30hubs since 2002 and the hubs are growing every year into various fields ranging from development of assistive technology to volcanlogy[1].

"HUBzero is now supported by a consortium including Purdue, Indiana, Clemson and Wisconsin. Researchers at Rice, the State University of New York system, the University of Connecticut and Notre Dame use hubs. Purdue offers a hub-building and -hosting service and the consortium also supports an open source release, allowing people to build and host their own"[**?**].

A hub web 2.0 functionality with an unique middleware, thereby providing a platform that is more powerful than an ordinary website. In a hub users will able to create, publish, share information, network and also have access to interactive visualization tools. The users of the hub ultimately collaborate to develop the educational community and make it a powerful resource. A hub can direct jobs to national resources such as TeraGrid, Purdue's DiaGrid as well as other cloud systems[2].

2. MIDDLEWARE

HUBzero mix of social networking and simulation is only possible with the help of unique middleware. HUBzero's middleware hosts the live session tools and makes it easy to connect the cloud computing infrastructure and supercomputing clusters to solve the large computational problems. The simulation tools runs on clusters of executions hosts hosted near the Web server and will be accessed by user's browser via VNC(Virtual Network Connection)[1].

Each user has a home directory which is unique to that particular user with access control, quota limitations and conventional ownership privileges. Each tool runs on a lightweight restricted virtual environment, that controls access to the networking, file systems and server processes. Tools are run with the privileges of a particular user[1].

Tools running on the hub will have a XII Window System environment. Each container runs a special X server which also acts as a VNC server is used to create the graphical session. The middleware of HUBzero controls the tool container's network operations. The middleware also monitors the start time as well as the duration of each connection. The connections that are not being used for a considerable period after starting is terminated and marked as inactive[3].

3. FEATURES

HUBzero unique blend of social networking and simulation power made it popular among research and scientific communities. HUBzero's platforms are sophisticated because of the various features it offers. HUBzero sites provides a lot of features for collaboration, development and deployment of tools, interactive user interfaces for simulation tools, online community groups, Wikis, Blogs, Provision for Feedback Mechanisms, Metrics for Usage, content management resources and everything that would be needed for a collaborative platform

3.1. Development and Deployment of Tools

Each hub comes with a companion site for developing source code based on open source package called Trac for project management. Uploading a tool is a little complicated. Tools must be uploaded, compiled, tested, fixed, compiled again, and tested

again many times before being published. Each tool will have its own project area within the site, with a repository for source code control, a ticketing system for bug tracking as well as a wiki area for project documentation.

A team member from development get access to workspace which is a Linux desktop running in a secured execution environment, accessed via a Web browser. The tools that are available on the hub dont come from the core development team rather they are developed by various collaborators across the world. Developers can use HUBzero's rapture toolkit to create a GUI with little effort[4].

3.2. Online Presentations

HUBzero facilitates Online Presentations along with the tools. The PowerPoint slides are combined with voice and animation. The online presentations provide user a standard seminar experience. HUBzero uses MacroMedia Breeze to deliver the online presentations in a compact format using Flash, which is present on 98 percent of the world desktops. The online presentations provided by HUBzero can also be distributed as podcasts which be accessed by the users on-the-go via their personal device assistants[4].

3.3. Interactive Simulation Tools

The underlying premise of HUBzero is its signature service to deliver live graphical simulation tools that are interactive and can be accessed using a normal web browser. The tools in a hub are interactive; you can zoom a molecule, analyze 3D volume interactively and need not even wait for the webpage to be refreshed. A user can visualize results in real-time and need for wait for processing. One can deploy new tools without having to write any special code for deploying on the web[4].

HUBzero's Rappture Toolkit helps to create GUI with little effort. The HUBzero infrastructure provides tool execution as well as delivery mechanism based on Virtual Network Computing(VNC). Using the architecture provided by HUBzero, the first ever developed hub, nanoHUB has brought about 50 simulation tools live in span of 2 years.

3.4. Collaboration and Support

HUBzero is a popular because it not only provides simulation tools, wikis etc., but also provides sophisticated mechanism for colleagues to collaborate among themselves and work together. HUBzero middleware hosts tool sessions, where a single session can be shared with among many number of people. Each session provides a textbox entry beneath it to allow the users to enter the colleagues names they wish to collaborate with. A group of people can see the same session at the same time and discuss ideas over the phone or instant messaging [4].

HUBzero also provides community forum modeled after Amazon.com's Askville where users can post questions. Users can also tag the posts with the tool names and concepts so that users can find questions matching their interests and post answers accordingly. Users also can file trouble reports if something goes wrong which are sent directly to tool development team. Developers view the help requests as tickets and investigate and resolve the issues. Users also can have access to wishlist where they can request new features for a particular tool. The developers can determine a wish's relative priority and judges which are important and requires little effort are ordered accordingly. There is points and bounty system which enhances involvement of users in the hub by providing premium access to the resources based on the levels of bounties and points.

3.5. Content Tagging, Wikis abd Blogs

The resources available on a hub are categorized by a series of tags. Each tag has an associated page on the hub where its resources are listed and meaning is defined. Tags can be defined by anyone like the contributor, users of the page or the hub adminstrator.

Hubs support topic pages. Each topic is created using a standard wiki syntax by a specified list of authors. Users can comment on the content of the wiki or even suggest changes. The original moderators of the topic are notified about the suggestions which they will consider if apt. The page can also be given a ownership. Topic pages act as lightweight articles that help to describe various resources on the hub[4].

4. APPLICATIONS

HUBzero platform has been very popular in science and engineering fields. Many hubs have come into existence in various fields ranging from cancer to nanotechnology. More than 30 hubs have been spawned since HUBzero platform had come into existence. All these hubs serve more than 8,50,000 visitors from 172 countries worldwide during year 2012 alone.

4.1. NanoHUB

nanoHUB is the reason behind the HUBzero platform. HUBzero was initially created to support nanoHUB.org, an online community for the Network for Computational Nanotechnology, which is funded by US National Science Foundation in 2002 to connect theorists who develop simulation tools with the experimenters and engineers.

nanoHUB is an online portal for nanotechnology where researchers, students and instructors collaborate to share scientific tools, simulation tools, educational material, research etc., It uses cyberinfrastructure to provide access to this tools and also the instructional materials. The users can run experiments, download lectures as well as review research[5].

4.2. Big Data Analysis in Social Science Using HUBzero

Datasets from social media are infact large and can grow beyond a individual or institution analytical capability of common software tools. Institutions such as Non-STEM or undergraduate schools lack the compute infrastructure and personnel needed to allows researchers, students, instructors to create and analyze large datasets.

The social science students are expected to be competent users. Therefore, in order to provide infrastructure necessary to expose undergraduates social science students to data intensive computing, State University of New York (SUNY) Oneonta teamed with University at Buffalo's Center for Computational Research (CCR) to establish a collaborative virtual community focusing on data intensive computing education[6].

5. LICENSE

HUBzero has been released as Open Source under the LGPL-3.0 license. HUBzero is community code unlike the GNU General Public License it doesn't "infect" your code. Under HUBzero's license, one can treat HUBzero as a "library," create your own unique components within our framework, and license your derivative works any way you like.

6. CONCLUSION

HUBzero has been a unique platform with social networking and simulation power has been able to resonate science and engineering communities. As hub keeps to grow the capabilities continues to grow, more tools and related content will be added. A hub lets various researchers, engineers collaborate to solve larger problems by connecting a series of models from independent authors.

REFERENCES

[1] M. McLennan and R. Kennell, "Hubzero: a platform for dissemination and collaboration in computational science and engineering," *Computing in Science & Engineering*, vol. 12, no. 2, 2010.

[2] M. McLennan and G. Kline, "Hubzero paving the way for the third pillar of science," webpage, 2011. [Online]. Available: https://www.hpcwire.com/2011/02/28/hubzero_paving_the_way_for_the_third_pillar_of_science/

[3] "Information technology at purdue research compting(rcac)," webpage. [Online]. Available: https://www.rcac.purdue.edu/services/hubzero/

[4] "hubzero:platform for scientific collaborations," webpage. [Online]. Available: https://hubzero.org/tour/features

[5] G. Klimeck, M. McLennan, S. P. Brophy, G. B. Adams III, and M. S. Lundstrom, "nanohub. org: Advancing education and research in nanotechnology," *Computing in Science & Engineering*, vol. 10, no. 5, pp. 17–23, 2008.

[6] J. M. Sperhac, S. Gallo, J. B. Greenberg, B. Lowe, B. Wilkerson, G. Fulkerson, and B. Heindl, "Teaching big data analysis in the social sciences using a hubzero-based platform," Oct 2014. [Online]. Available: https://hubzero.org/resources/1235

Apache Flink: Stream and Batch Processing

JIMMY ARDIANSYAH[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *jardians@indiana.edu - S17-IR-2002*

Research Article-02, April 30, 2017

Apache Flink is an open-source system for processing streaming and batch data. Flink is built on the philosophy that many classes of data processing applications, including real-time analytics, continuous data pipelines, historic data processing (batch), and iterative algorithms can be expressed and executed as pipelined fault-tolerant dataflows.

Keywords: Apache Flink, Batch Data Processing, Stream Data Processing

https://github.com/jardians/sp17-i524/blob/master/paper2/S17-IR-2002/report.pdf

1. INTRODUCTION

Data stream and batch data processing were traditionally considered as two very different types of applications. They were programmed using different programming models and APIs, and were executed by different systems. Normally, batch data analysis made up for the biggest share of the use cases, data sizes, and market, while streaming data analysis mostly served specialized applications.

These continuous streams of data come for example from web log files, application log files, and databases log files. Rather than treating the streams as streams, today's setups ignore the continuous and timely nature of data production. Instead, data records are batched into static data sets (hourly, daily, or monthly) and then processed in a time-based fashion. Data collection tools, workflow managers, and schedulers orchestrate the creation and processing of batches, in what is actually a continuous data processing pipeline. Apache Flink follows a paradigm that embraces data stream processing as the unifying model for real time analysis, continuous streams, and batch processing both in the programming model and in the execution engine.

Flink programs can compute both data stream and batch data accurately that avoiding the need to combine different systems for the two use cases. Flink also supports different notions of time (event-time, ingestion-time, processing-time) in order to give programmers high flexibility in defining how events should be correlated. [1].

2. HISTORY DEVELOPMENT

Flink has its origins in the Stratosphere project, a research project conducted by three Berlin-based Universities as well as other European Universities between 2010 and 2014. The project had already attracted a broader community base such as NoSQL and Big Data Developers Groups. This strong community base is one reason the project was appropriate for incubation under the Apache Software Foundation (ASF).

A fork of the Stratosphere code was donated in April 2014 to the Apache Software Foundation (ASF) as an incubating project with an initial set of committers consisting of the core developer of the system. Shortly thereafter, many of the founding committers left university to start a company to commercialize Flink such as Data Artisans.

During incubation, the project name had to be changed from Stratosphere because of potential confusion with an unrelated project. The name Flink was selected to honor the style of this stream and batch processor. In German, the word "Flink" means fast or agile. A logo showing a colorful squirrel was chosen because of squirrel are fast and agile. The project completed incubation quickly, and in December 2014, Flink graduated to become a top-level project of the Apache Software Foundation (ASF). Flink is one of the 5 largest big data projects of Apache Software Foundation (ASF) with a community of more than 200 developers across the globe and several production installations in Fortune Global 500 companies. In Octorber2015, the Flink project held its first annual conference in Berlin called Flink Forward [2] [3].

3. DESIGN

The core computational fabric of Flink (labeled as "Flink runtime" in the Figure-1) is a distributed system that accept streaming dataflow programs and executes them in a fault-tolerant manner in one or more machines. This runtime can run in a cluster as an application of YARN (Yet Another Resources Negotiator) or within a single machine which is very useful for debugging Flink applications.

The program accepted by the runtime are very powerful, but are verbose and difficult to program directly. For that reason, Flink offer developer-friendly APIs that layer on the top of the runtime and generate there streaming dataflow programs. Apache Flink includes two core APIs: a DataStream API for bounded or unbounded streams of data and a DataSet API for bounded data sets. Flink also offers a Table API, which is a SQL-like expression language for relational stream and batch processing that can be easily embedded in Flink's DataStream and DataSet APIs. The highest-level language supported by Flink is SQL, which is semantically similar to the Table API and represents programs as SQL query expressions [4].

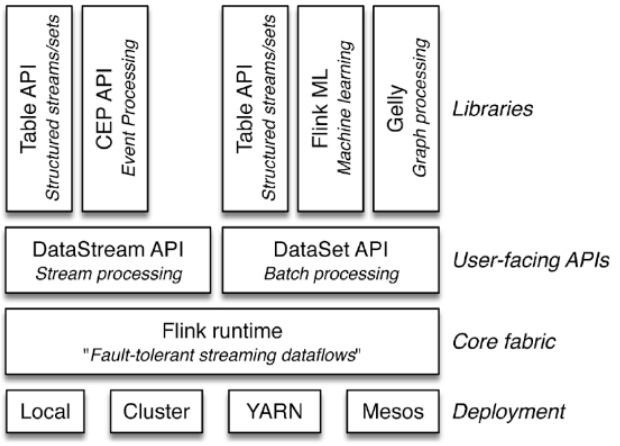


Fig. 1. Key concept of Flink Stack [4]

3.1. Data Stream API

DataStream programs in Flink are regular programs that implement transformations on data streams (e.g., filtering, updating state, defining windows, aggregating). The data streams are initially created from various sources (e.g., message queues, socket streams, files). Results are returned via sinks, which may for example write the data to files, or to standard output (for example the command line terminal). Flink programs run in a variety of contexts, standalone, or embedded in other programs. The execution can happen in a local JVM, or on clusters of many machines. The DataStream API includes more than 20 different types of transformations and is available in Java and Scala languages. A simple example of a stateful stream processing program is an application that emits a word count from a continuous input stream and groups the data in 5-second windows.

```scala
import org.apache.flink.api.scala._

object WordCount {
  def main(args: Array[String]) {

    val env = ExecutionEnvironment.getExecutionEnvironment
    val text = env.fromElements(
      "Who's there?",
      "I think I hear them. Stand, ho! Who's there?")

    val counts = text.flatMap { _.toLowerCase.split("\\W+") filter { _.nonEmpty } }
      .map { (_, 1) }
      .groupBy(0)
      .sum(1)

    counts.print()
  }
}
```

Fig. 2. Scala Example Program: Counts the words coming from a web socket in 5 second windows [4]

3.2. DataSet API

DataSet programs in Flink are regular programs that implement transformations on data sets (e.g., filtering, mapping, joining, grouping). The data sets are initially created from certain sources (e.g., by reading files, or from local collections). Results are returned via sinks, which may for example write the data to (distributed) files, or to standard output (for example the command line terminal). Flink programs run in a variety of contexts, standalone, or embedded in other programs. The execution can happen in a local JVM, or on clusters of many machines.

```scala
import org.apache.flink.streaming.api.scala._
import org.apache.flink.streaming.api.windowing.time.Time

object WindowWordCount {
  def main(args: Array[String]) {

    val env = StreamExecutionEnvironment.getExecutionEnvironment
    val text = env.socketTextStream("localhost", 9999)

    val counts = text.flatMap { _.toLowerCase.split("\\W+") filter { _.nonEmpty } }
      .map { (_, 1) }
      .keyBy(0)
      .timeWindow(Time.seconds(5))
      .sum(1)

    counts.print

    env.execute("Window Stream WordCount")
  }
}
```

Fig. 3. Scala Example Program: WordCount [4]

3.3. Table API

The Table API is a declarative DSL centered around tables, which may be dynamically changing tables (when representing streams). The Table API follows the (extended) relational model: Tables have a schema attached (similar to tables in relational databases) and the API offers comparable operations, such as select, project, join, group-by, aggregate, etc. Table API programs declaratively define what logical operation should be done rather than specifying exactly how the code for the operation looks.

Though, the Table API is extensible by various types of user-defined functions, it is less expressive than the Core APIs, but more concise to use (less code to write). In addition, Table API programs also go through an optimizer that applies optimization rules before execution [5]. One can seamlessly convert between tables and DataStream/DataSet, allowing programs to mix Table API and with the DataStream and DataSet APIs. The highest level abstraction offered by Flink is SQL. This abstraction is similar to the Table API both in semantics and expressiveness, but represents programs as SQL query expressions. The SQL abstraction closely interacts with the Table API, and SQL queries can be executed over tables defined in the Table API.

4. IMPLEMENTATION OF APACHE FLINK

4.1. Alibaba

[3]This huge e-commerce group works with buyer and suppliers via its web portal. The company's online recommendation are produced by Flink. One of the attractions of working with true streaming engines such as Flink is that purchases that are being made during the day can be taken into account when recommending products to users. This particularly important on special day (holidays) when the activities is unusually high. This is an example of a use case where efficient stream processing is a big advantage over batch processing.

4.2. Otto Group

[3]The Otto is the world's second-largest online retailer in fashion and lifestyle in Europe. Otto had resorted to developing its own streaming engine because when it first evaluate the open source options, it could not find one that fit its requirement. After testing Flink, Otto found it fit their needs for streaming processing which include crowd-sourced user-agent identification, and a search session identifier.

5. CONCLUSION

Flink is not the only technology available to work with streaming and batch processing. There are a number of emerging technologies being developed and improved to address these needs. Obviously people choose to work with a particular technology for a variety of reason, but the strengths of Flink, the ease of working with it, and the wide range of ways it can be used to advantage make it an attractive option. That

6. ACKNOWLEDGEMENTS

This work was done as part of the course "I524: Big Data and Open Source Software Projects" at Indiana University during Spring 2017

REFERENCES

[1] A. Katsifodimos and S. Schelter, "Apache flink: Stream analytics at scale," in *IEEE International Conference on Cloud Engineering Workshop (IC2EW)*, 2016, pp. 193–193. [Online]. Available: http://ieeexplore.ieee.org/document/7527842

[2] wikipedia, "Apache flink," Web Page, Mar. 2017, accessed: 2017-03-20. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Flink

[3] E. Friedman and K. Tzoumas, *Introduction to Apache Flink: Stresam Processing for Real Time and Beyond*. Sebastopol California: OReilly Media Inc, 2016.

[4] Apache Software Foundation, "Introducing flink streaming," Web Page, Mar. 2017, accessed: 2017-03-22. [Online]. Available: https://flink.apache.org/news/2015/02/09/streaming-example.html

[5] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas, "Apache flink: Stream and batch processing in a single engine," *IEEE Data Eng. Bull.*, vol. 38, pp. 28–38, 2015. [Online]. Available: https://www.researchgate.net/publication/308993790_Apache_Flink_Stream_and_Batch_Processing_in_a_Single_Engine

Jelastic

AJIT BALAGA, S17-IR-2004[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: abalaga@iu.edu, ajit.balaga@gmail.com*

project-000, April 30, 2017

Jelastic (acronym for Java Elastic) is an unlimited PaaS and Container based IaaS within a single platform that provides high availability of applications, automatic vertical and horizontal scaling via containerization to software development clients, enterprise businesses, DevOps, System Admins, Developers, OEMs and web hosting providers.[1]

Keywords: Cloud, I524

https://github.com/argetlam115/sp17-i524/blob/master/paper2/S17-IR-2004/report.pdf

1. INTRODUCTION

Enterprises are moving away from traditional virtualization solutions and transitioning into the cloud. As software development in the enterprise becomes more agile, there is an equivalent demand on IT to provide an infrastructure that is responsive, scalable, highly available and secure. Enterprise IT departments are responding with private cloud and hybrid cloud solutions that provide IT-as-a-Service, a utility approach that delivers an agile infrastructure to the user community, with ultimate control for administrators but self-management for developers and users where appropriate. The promise of the cloud was: simplicity, scalability, availability and reduced operating cost. However, enterprises are quickly finding that current large-scale cloud implementations are often complicated and expensive, often requiring the help of third party integrators. Jelastic is a cloud service that solves the above problems the above promises and allows enterprises to speed up the development process.This paper introduces the architecture behind Jelastic.[2]

2. JELASTIC

Jelastic is a cloud platform solution that combines benefits of both Platform as a Service (PaaS) and Container as a Service cloud models, using an approach unleashes the full potential of a cloud for enterprises, ISVs, hosting service providers and developers. Jelastic software encompasses PaaS functionality, a complete infrastructure, smart orchestration, and containers support - all together.[3]

Jelastic solutions benefits for all kinds of clients:

- enterprises
- hosting providers
- developers

[3]

Some of the Jelastic key benefits:

- a turnkey Platform for Public, Private, Hybrid and Multi-Cloud deployments with automated continuous integration, delivery and upgrade processes
- support of numerous software stacks, extended with cartridges packaging model and custom Docker containers[1]
- automated replication and true automated scaling, both vertical and horizontal - all applications scale up and down on demand
- various development environments for the most comfortable work experience - intuitive UI, open API and SSH access to containers
- intelligent workloads distribution with multi-cloud and multi-region management[4]
- smart pricing integration - alongside multiple billing systems support, Jelastic provides comprehensive billing engine, quotas and access control policies
- embedded troubleshooting tools for metering, monitoring, logging, etc.

[3]

3. ARCHITECTURE

A consistent outline of the underlying Jelastic components with pointers to the corresponding documentation, namely:[5]

3.1. Cloudlet

Cloudlet is a special infrastructure component that equals to 128 MiB of RAM and 400 MHz of CPU power simultaneously. Such high granularity of resources allows the system to allocate the exactly required capacity for each instance in the environment. There are two types of cloudlets:

- Reserved Cloudlets are fixed amount of resources reserved in advance. Reserved cloudlets are used when the application load is permanent.
- Dynamic Cloudlets are added and removed automatically according to the amount of resources required. Dynamic cloudlets are used for applications with variable load or when it cannot be predicted in advance.

[3]

3.2. Container

Container (node) is an isolated virtualized instance, provisioned for software stack handling and placed on a particular hardware node. Each container can be automatically scaled, both vertically and horizontally, making hosting of applications truly flexible. The platform provides certified containers for a lot of commonly used languages and the ability to deploy custom Docker containers. Each container has its own private IP and unique DNS record.[6]

3.3. Layer

Layer (node group) is a set of similar containers in a single environment. There is a set of predefined layers within Jelastic topology wizard for certified containers, such as:[7]

- load balancer (LB)
- compute (CP)
- database (DB)
- data storage (DS)
- cacheVPS
- build node
- extra (custom layer)

[3]

The layers are designed to perform different actions with the same type of containers at once. The nodes can be simultaneously restarted or redeployed, as well as horizontally scaled manually or automatically based on the load triggers, checked for errors in the common logs and stats and make the required configurations via file manager for all containers in a layer. The containers of one layer are distributed across different hardware servers.[8]

3.4. Environment

Environment is a collection of isolated containers for running particular application services. Jelastic provides built-in tools for convenient environment management. There is a number of actions that can be performed for the whole environment, such as stop, start, clone, migrate to another region, share with team members for collaborative work, track resource consumption, etc. Each environment has its own internal 3rd level domain name by default. A custom external domain can be easily bound or even further swapped with another environment for traffic redirection.[1]

3.5. Application

Application is a combination of environments for running one project. A simple application with one or two stacks can be run inside a single environment. Applications with more complex topology usually require more flexibility during deploy or update processes They may be distributed across different types of servers and several environments, to be maintained independently. Application source code can be deployed from:[4]

- GIT/SVN repository
- local archive
- custom Docker template

[3]

3.6. Hardware Node

Hardware node is a physical server or a big virtual machine that is virtualized via KVM, ESXi, Hyper-V, etc. Hardware nodes are sliced into small isolated containers that are used to build environments. Such partition provides the industry-leading multitenancy, as well as high density and smart resource utilization with the help of containers distribution according to the load across hardware nodes.[5][7]

3.7. Environment Region

Environment region is a set of hardware nodes orchestrated within a single isolated network. Each environment region has its own capacity in a specific data centre, predefined pool of private and public IP addresses and [4] corresponding resource pricing. Moreover, the initially chosen location can be effortlessly changed by migrating the project between available regions.[6]

3.8. Jelastic Platform

Jelastic Platform is a group of environment regions and cluster orchestrator to control and act like a single system. This provides versatile possibilities to develop, deploy, test, run, debug and maintain applications due to the multiple options while selecting hardware - different capacity, pricing, location, etc. The platform provides a multi-data center or even multi-cloud solution for running your applications within a single panel, where each Platform is maintained by a separate hosting service provider with its local support team.[3]

4. CONCLUSION

For enterprises, moving from traditional virtualization to the cloud using PaaS and IaaS can be a daunting proposition. However, the cloud market is growing rapidly and enterprises are recognizing that PaaS allows them to develop and deploy scalable, highly available cloud-based applications in a rapid and agile fashion. Enterprises can capitalize on this new and sticky revenue stream by quickly implementing PaaS and establishing a brand-defining presence in the market. Jelastic provides the only integrated private cloud solution that integrates PaaS/IaaS and is specifically built for enterprises.[2]

REFERENCES

[1] "Jelastic," Web page, Dec. 2016, page Version ID: 754931676. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Jelastic&oldid=754931676

[2] “5 Key Reasons Why Jelastic PaaS and IaaS for Private Cloud is Better,” Web page, Jul. 2014. [Online]. Available: http://blog.jelastic.com/2014/07/10/why-you-should-use-jelastic-platform-as-infrastructure-for-private-cloud/
[3] “Jelastic Multi-Cloud PaaS and CaaS for Business,” Web page. [Online]. Available: https://jelastic.com/
[4] S. Yangui and S. Tata, “Cloudserv: Paas resources provisioning for service-based applications,” in *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*. IEEE, 2013, pp. 522–529. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6531799/
[5] “Jelastic Cloud Union Catalog: Choose Your Service Provider,” Web page. [Online]. Available: https://jelastic.cloud/
[6] “Auto scaling Jelastic PaaS in UK, USA Southwest, and Singapore,” Web page. [Online]. Available: https://www.layershift.com/jelastic
[7] J. Chavarriaga, C. A. Noguera, R. Casallas, and V. Jonckers, “Architectural tactics support in cloud computing providers: the jelastic case,” in *Proceedings of the 10th international ACM Sigsoft conference on Quality of software architectures*. ACM, 2014, pp. 13–22. [Online]. Available: http://dl.acm.org/citation.cfm?id=2602580
[8] “Jelastic PaaS Cloud Application Hosting,” Web page. [Online]. Available: https://www.lunacloud.com/cloud-jelastic

An Overview of Apache Spark

SNEHAL CHEMBURKAR[1] AND RAHUL RAGHATATE[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
* Corresponding authors: snehchem@iu.edu, rraghtate@iu.edu

paper-1, April 30, 2017

Apache Spark, developed at UC Berkeley AMPLAB, is a high performance framework for analyzing large datasets [1]. The main idea behind the development of Spark was to create a generalized framework that could process diverse and distributed data as opposed to MapReduce which only support batch processing of data. Spark has multiple libraries built on top of its core computational engine which help process diverse data. This paper will discuss the Spark runtime architecture, its core and libraries.

Keywords: Spark, RDDs, DAG, Spark SQL, MLlib, GraphX, I524

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IR-2006/report.pdf

1. INTRODUCTION

Spark is an open source, easy-to-use distributed cluster computing engine for processing the different types of data available these days. It was built with a view to overcome the shortcomings of MapReduce. Spark provides a generalized framework which can efficiently process MapReduce jobs (batch processing) as well as iterative algorithms, interactive data mining and streaming analytics. Iterative algorithms include many machine learning algorithms which iterate over the same dataset to optimize a parameter. Interactive data mining refers to executing ad-hoc queries to explore the dataset.

MapReduce can process iterative algorithms by splitting each iteration into a separate MapReduce job. Each job must then read data from a stable storage and write it back to a stable storage at each intermediate step. This repeated access to the stable storage systems like physical disks or HDFS increases the processing time while reducing the efficiency of the system. For interactive data mining in Hadoop, the data is loaded in memory across a cluster, and queried repeatedly. Each query is executed as separate MapReduce job which reads data from the HDFS or hard drives thus incurring significant latency (tens of seconds) [2]. Spark is specialized to make data analysis faster in terms of data write speed as well as program execution. It supports in-memory computations which enable faster data querying compared to disk-based systems like Hadoop.

Spark is implemented in Scala, a high level programming language that runs on JVM. It makes programming easier by providing a clean and concise API for Scala, Java and Python [2]. Spark also provides libraries that allow for iterative, interactive, streaming and graph processing. Spark SQL library provides for interactive data mining in Spark. MLlib provides Spark with machine learning algorithms required for iterative computations. Similarly Spark streaming and GraphX libraries enable Spark to process real-time and graph processing data respectively. These high level components required for processing the diverse workloads such as structured or streaming data are powered by the Spark Core. The distribution, scheduling and monitoring of clusters is done by the Spark Core.

The Spark Core and it's higher level libraries are tightly integrated meaning when updates or improvements are implemented in the Spark Core help improve the Spark libraries as well. This makes it easier to write applications combining different workloads. This is explained nicely in the following example. One can build an application using machine learning libraries to process real time data from streaming sources and analysts can simultaneously access the data using SQL in real time. In this example three different workloads namely SQL, streaming data and machine learning algorithms can be implemented in a single system which is a requirement in today's age of big data.

2. SPARK COMPONENTS

Figure 1 depicts the various building blocks of the Spark stack. Spark Core is the foundation framework that provides basic I/O functionality, distributed task scheduling and dispatching [1]. It is the core computational engine of the system. Resilient Distributed Datasets (RDD) and Directed Acyclic Graphs (DAG) are two important concepts in Spark. RDDs are a collection of read-only Java or Python objects parallelized across a cluster. DAGs, as the name suggests are directed graphs with no cycles. The libraries or packages supporting the diverse workloads, built on top of the Spark Core, include Spark SQL, Spark Streaming, MLlib (machine learning library) and GraphX. The Spark Core runs atop cluster managers which are covered in section 4.

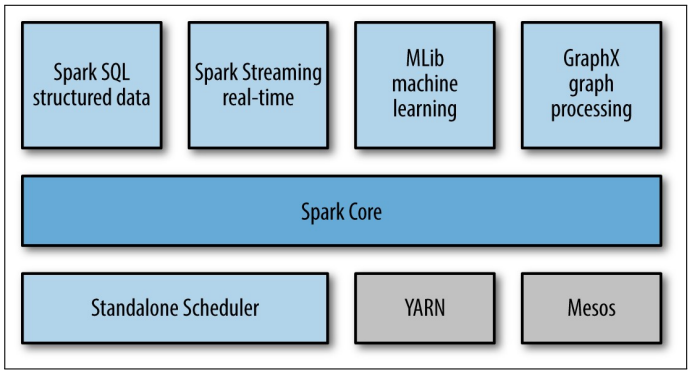


Fig. 1. Basic Components of Spark [3].

2.1. Resilient Distributed Datasets (RDD)

Resilient Distributed Datasets (RDD) [4] are Spark's primary abstraction, which are a fault-tolerant collection of elements that can be operated in parallel. RDDs are immutable once they are created but they can be transformed or actions can be performed on them [1]. Users can create RDDs through external sources or by transforming another RDD. Transformations and Actions are the two types of operations supported by RDDs.

- *Transformations*: Since RDDs are immutable, the transformations return a new RDD and not a single value. RDDs are lazily evalutated i.e they are not computed immediately when a transformation command is given. They wait till an action is received to execute the commands. This is called Lazy evaluation. Examples of transformation functions are map, filter, ReduceByKey, FlatMap and GroupByKey [1].
- *Actions* are operations that result in a return value after computation or triggers a task in response to some operation. Some Action operations are first, take, reduce, collect, count, foreach and CountByKey [1].

RDDs are ephemeral disk, which means they do not persist data. However, users can explicitly persist RDDs to ease data reuse. Traditional distributed computing systems provide fault tolerance through checkpoint or data replication. RDDs provide fault-tolerance through lineage. The transformations used to build a data set are logged and can be used to rebuild the original data set through its lineage [4]. If one of the RDD fails, it has enough information about of its lineage so as to recreate the dataset from other RDDs, thus saving cost and time.

2.2. Directed Acyclic Graphs

Directed Acyclic Graph (DAG), which supports acyclic data flow, "consists of finitely many vertices and edges, with each edge directed from one vertex to another, such that there is no way to start at any vertex v and follow a consistently-directed sequence of edges that eventually loops back to v again [5]." When we run any application in Spark, the driver program converts the transformations and actions to logical directed acyclic graphs (DAG). The DAGs are then converted to physical execution plans with a set of stages which are distributed and bundled into tasks. These tasks are distributed among the different worker nodes for execution.

2.3. Spark SQL

Spark SQL [3] is a library built on top of the Spark Core to support querying structured data using SQL or Hive Query Language. It allows users to perform ETL (Extract, Transform and Load) operations on data from various sources such as JSON, Hive Tables and Parquet. Spark SQL provides developers with a seamless intermix of relational and procedural API, rather than having to choose between the two. It provides a DataFrame API that enables relational operations on both the in-built collections as well as external data sources. Spark SQL also provides a novel optimizer called Catalyst, to support the different data sources and algorithms found in big data [6].

2.4. Spark Streaming

Spark Streaming [3] library enables Spark to process real time data. Examples of streaming data are messages being published to a queue for real time flight status update or the log files for a production server. Spark's API for manipulating data streams is very similar to the Spark Core's RDD API. This similarity makes it easier for users to move between projects with stored and real-time data as the learning curve is short. Spark Streaming is designed to provide the same level of fault tolerance, throughput and scalability as the Spark Core.

2.5. MLlib

MLlib [3] is a rich library of machine learning algorithms for, which can be accessed from Java, Scala as well as Python. It provides Spark with various machine learning algorithms such as classification, regression, clustering, and collaborative filtering. It also provides machine learning functionality such as model evaluation and data import. The common machine learning algorithms include K-means, navie Bayes, logistic regression, principal component analysis and so on. It also provides basic utilities for feature extractions, optimizations and statistical analysis to name a few [7].

2.6. GraphX

GraphX is a graph processing framework built on top of Spark. ETL, exploratory data analysis and iterative graph computations are unified within a single systems using GraphX [8]. It introduces the Resilient Distributed Property Graph, which is directed multi-graph having properties attached to each edge and vertex [9]. GraphX includes a set of operators like aggregateMessages, subgraph and joinVertices, and an optimized variant of Pregel API [8]. It also includes builders and graph algorithms to simplify graph analytics tasks [1].

3. RUNTIME ARCHITECTURE

The runtime architecture of Spark, illustrated in Figure 2. It consists of a driver program, a cluster manager, workers and the HDFS (Hadoop Distributed File System) [1]. Spark uses a master/slave architecture in which the driver program is the master whereas the worker nodes are the slaves. The driver runs the main() method of the user program which creates the SparkContext, the RDDs and performs transformations and actions [3].

When we launch an application using the Spark Shell it creates a driver program which in turn initializes the SparkContext. Each Spark application has its own SparkContext object which is responsible for the entire execution of the job. The SparkContext object then connects to cluster manager to request resources for its workers. The cluster manager provide executors to worker nodes, which are used to run the logic and also store the application data. The driver will send the tasks to the executors based on the data placement. The executors register themselves with

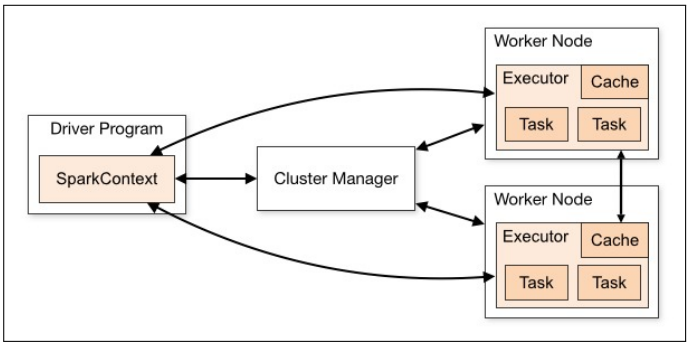


Fig. 2. Runtime Architecture of Spark [10].

the driver, which helps the driver keep tabs on the executors. Driver can also schedule future tasks by caching or persisting data.

4. DEPLOYMENT MODES

Spark can be deployed in local and clustering modes. The local mode of Spark runs a single node. In clustering mode, Spark can connect to any of the following cluster managers - standalone cluster manager, YARN or Mesos - explained in the following sections.

4.1. Standalone

Standalone cluster manager is the built-in cluster manager provided by Spark in its default distribution. The Standalone Master acts as the resource manager and allocates resources to the Spark application based on the number of cores. Spark standalone mode is not very popular in production environments due to reliability issues [11]. To run your Spark application in a standalone clustered environment, make sure that Spark must be installed on all nodes in the cluster. Once Spark is available on all nodes, follow the steps given in Spark documentation [12] to start the master and workers. The master server has a web UI which is located at `http://localhost:8080/` by default. This UI will give information regarding the number of CPUs and memory allocated to each node [12].

4.2. YARN

YARN (Yet Another Resource Negotiator) is the resource manager in Hadoop ecosystem. Like the Standalone cluster master, the YARN ResourceManager decides which applications get to run executor processes, where they run it and when they run it. The YARN NodeManager is the slave service that runs on every node and runs the executor processes. This service also helps monitor the resource consumption at each node. YARN is the only cluster manager for Spark that provides security support [11]. To run Spark on YARN, Spark distribution with YARN support must be downloaded from the Apache Spark download page.

4.3. Apache Mesos

Apache Mesos is an open source distributed systems kernel, using principles similar to the Linux kernel but at a different level of abstraction [13]. This kernel provides Spark with APIs for resource management and scheduling across the cluster and runs on each node in the cluster. While scheduling tasks, Mesos considers the other frameworks that may coexist on the same cluster. The advantage of deploying Spark with Apache Mesos include dynamic partitioning between Spark and other frameworks and scalable partitioning between multiple instances of Spark. Installation of Mesos for Spark is similar to its installation for use by any other frameworks. You can either download Mesos release from its source or from the binaries provided by third party projects like Mesosphere [11].

5. EASY INSTALLATION USING PRE-BUILT PACKAGES

To run Spark on your Windows, Linux or Mac systems you need to have Java 7+ installed on your system. To verify if you have java installed on your Linux machine type `java -version` command in the terminal. If you do not have Java installed on your system you can download it from the Oracle website [14]. The environment variable `PATH` or `JAVA_HOME` must be set to point to the Java installation. Now that the Java prerequisite is satisfied, go to the download page of Apache Spark, select the 2.1.0 (latest version as on 26th March 2017) version of Spark, select the pre-built Hadoop 2.7 package and download the `spark-2.1.0-bin-hadoop2.7.tgz` file. Then, go to the terminal and change the directory folder to where the file is located and execute the following command to unzip the file

```
tar -xvf spark-2.1.0-bin-hadoop2.7.tgz -C -/
```

This will create a folder `spark-2.1.0-bin-hadoop2.7` in that directory. Move this folder to the `/usr/local/spark` using command `mv spark-2.1.0-bin-hadoop2.7 /usr/local/spark` To set the environment variable for Spark, open the `.bashrc` file using command `sudo nano /.bashrc` and add the following lines at the end of this file.

```
export SPARK_HOME = /usr/local/spark
export PATH=$PATH:$SPARK_HOME/bin
```

Go back to your home directory and execute `.bashrc` using command `. .bashrc` for the changes to take effect. This change can be verified by executing command `echo $PATH`. The `PATH` variable should now reflect the path to the spark installation. To verify that Spark is installed correctly, execute command `$spark-shell` in the terminal. It will display the Spark version and then enter the Scala prompt.

6. BUILDING SPARK BASED APPLICATIONS

The first step to start building Spark based applications is exploring the data in Spark Local Mode and developing a prototype [15]. Spark local mode runs on a single node and can be used by developers to learn Spark by building a sample application that leverages the functionalities of Spark API. The developer can use Spark Shells like Scala REPL or PySpark to develop a quick prototype. It can then be packaged as a Spark application using Maven or Scala Build Tool(SBT) [15]. The second step involves deploying the Spark application to production. To achieve this, the developer will fine tune the prototype by running it against a larger dataset. This involves running Spark in cluster mode on YARN or Mesos. Thus the prototype application created in the local mode of Spark will now be submitted as a Spark job to the production cluster [15].

7. PERFORMANCE MONITORING TOOLS

Spark provides a web interface to monitor its applications. By default, each SparkContext launches a webUI, at port 4040 [16]. This UI displays the the memory usage statistics, list of scheduler stages and tasks, environmental information and information

about the executors. This interface can be access by opening `http://<driver-node>:4040` in the web browser [16]. If multiple instances of SparkContext are running on the same machine, then they will bind to successive ports beginning with 4040 (4041, 4042, 4043, ...) [16]. This information is only available for the life of the application. To view this information after the life of the application, set `spark.eventLog.enabled` to true before starting the application. This will configure Spark to store the event log to persistent storage [16].

A REST API enables the metrics to be extracted in JSON format, making it easier for developers to create visualizations and monitoring tools for Spark [16]. These metrics can also be extracted as HTTP, JMX, and CSV files by configuring the metric system in the configuration file present at `$SPARK_HOME/conf/metrics.properties`. In addition to these, external tools like Ganglia, dstat, iostat, iotop, jstack, jmap, jmap, and jconsole can also be monitor Spark performance [16].

8. USE CASES

In its early days, Spark was adopted in production systems by companies like Yahoo, Conviva, and ClearStory for personlization, analytics, streaming and interactive processing. These use cases are explained in further paragraphs.

Yahoo News Personalization: This project implements machine learning algorithms on Spark to improve news personalization for their visitors. Spark runs on Hadoop Yarn to use existing data and clusters. In order to achieve personalization, the system will learn about users' interests from their clicks on the web page. It also needs to learn about each news and categorize it. The SparkML algorithm written for this project was 120 lines of Scala code as compared to the 15,000 lines of C++ code used previously [17].

Yahoo Advertisement Analytics: In this project Yahoo leverages Hive on Spark to query and visualize the existing BI analytic data that was stored in Hadoop. Since Hive on Spark (Shark) uses the standard Hive server API, any tools that can be plugged into Hive, will automatically work with Shark. Thus visualization tools like Tableau that are compatible with Hive can be used with Shark to interactively query and view their ad visit data [17].

Monitor Network Conditions in Real-time: Conviva is a video streaming company with a huge video feed database. To ensure quality service, it requires pretty sophisticated technology to be applied behind the scenes to ensure high quality service. With the increase in internet speeds, people's tolerance towards buffering or delays has plummeted. To keep up with the rising expectations of high quality and speed for streaming videos, Conviva implemented Spark Streaming to learn about the network conditions in real-time. This information is then feed to the video player running on the user's laptop to optimize the video speeds [17].

Merge Diverse Data Sources: ClearStory develops data analytics software with speciality in data harmonization. To merge data from internal and external sources for its business users, they turned to Spark, which is one of the core components of their interactive and real-time product [17].

Credit Card Fraud Detection: Using Spark Streaming on Hadoop, banks can detect fraudulent transactions in real-time. The incoming transactions are verified in real-time against a known database of fraudulent transactions. Thus a match against the known database will alert the call center personnel to instantly verify the transaction with the credit card owner. The authentic transactions are stored to the Hadoop file system where they are used to continuously update the model using deep machine learning techniques [18].

Network Security: Spark can be used to examine network data packets for traces of malicious activity. Spark streaming checks the data packets against known threats and then forwards the unmatched data packets to the storage devices where it is further analyzed using the GraphX and MLlib libraries [18].

Genomic Sequencing: Genomic companies are leveraging the power of Spark to align chemical compounds with genes. Spark has reduced the genome data processing time from a few weeks to a couple of hours [18].

These are few of the real-world use cases of Spark. Real-world applications of Spark that incorporate MongoDB are Content Recommendations, Predictive Modeling, Targeted Ads and Customer Service [19].

9. WHEN NOT TO USE SPARK

Apache Spark is not the most suitable data analysis engine when it comes to processing (1) data streams where latency is the most crucial aspect and (2) when the available memory for processing is restricted. In cases where latency is the most crucial aspect we can get better results using Apache Storm. Since Spark maintains it's operations in memory, Hadoop MapReduce should be preferred, when available memory is restricted [20].

10. EDUCATIONAL RESOURCES

The Apache Spark website has a detailed documentation on the how to get started with Spark [21]. It explains the concepts and shows examples to help us familiarize with Spark.

11. LICENSING

Apache Spark is an open-source software licensed under the Apache License 2.0 [22]. Under this license, it is free to download and use this software for personal or commercial purposes. It forbids the use of marks owned by the Apache Software Foundation in a way that might imply that you are the creator of the Apache Software. It requires that you copy the license in any redistribution made by you which includes the Apache Software. You need to provide acknowledgement for any distributions that include the Apache Software [22].

12. CONCLUSION

Apache Spark is an open source cluster computing framework, which has emerged as the next generation big data processing engine surpassing Hadoop MapReduce. Spark facilitates in-memory computations which help execute the diverse workloads efficiently. It's ability to join datasets across various diverse data sources is one of it's major attributes. As mentioned in the previous section, Apache Spark is suitable for almost any kind of big data analysis except for the following scenarios: (1) where latency is the most crucial aspect and (2) when the available memory for processing is restricted. Spark finds place in almost all types of big data analysis projects, as seen from the wide range of use cases, due to it's core features (RDDs and in-memory computation) and different libraries.

ACKNOWLEDGEMENTS

This paper is written as part of the I524: Big Data and Open Source Software Projects coursework at Indiana University. We

would like to thank our Prof. Gregor von Laszewski, Prof. Gregory Fox and the AIs for their help and support

REFERENCES

[1] A. Bansod, "Efficient big data analysis with apache spark in hdfs," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 4, no. 6, pp. 313–316, Aug. 2015.

[2] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, ser. HotCloud'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 10–10. [Online]. Available: https://www.usenix.org/legacy/event/hotcloud10/tech/full_papers/Zaharia.pdf

[3] H. Karau, A. Konwinski, P. Wendell, and M. Zaharia, *Learning Spark: Lightning-Fast Big Data Analytics*, 1st ed. O'Reilly Media, Inc., Feb. 2015. [Online]. Available: https://www.safaribooksonline.com/library/view/learning-spark/9781449359034/

[4] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2012, San Jose, CA, USA, April 25-27, 2012*. Berkeley, CA, USA: USENIX Association, 2012, pp. 15–28. [Online]. Available: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/zaharia

[5] Wikipedia, "Directed acyclic graph - wikipedia," Web Page, Dec. 2016, accessed: 02-26-2017. [Online]. Available: https://en.wikipedia.org/wiki/Directed_acyclic_graph

[6] M. Armbrust, R. S. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, M. J. Franklin, A. Ghodsi, and M. Zaharia, "Spark sql: Relational data processing in spark," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '15. New York, NY, USA: ACM, 2015, pp. 1383–1394. [Online]. Available: http://doi.acm.org/10.1145/2723372.2742797

[7] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar, "Mllib: Machine learning in apache spark," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, Jan. 2016. [Online]. Available: http://www.jmlr.org/papers/volume17/15-237/15-237.pdf

[8] Apache Spark Foundation, "Graphx | apache spark," Web Page, accessed: 04-09-2017. [Online]. Available: http://spark.apache.org/graphx/

[9] S. Gupta, *Learning Real-time Processing with Spark Streaming*. Birmingham, UK: Packt Publishing Ltd, Sep. 2015.

[10] Apache Software Foundation, "Cluster mode overview - spark 2.1.0 documentation," Web Page, accessed: 02-22-2017. [Online]. Available: http://spark.apache.org/docs/latest/cluster-overview.html

[11] E. Chan, "Configuring and deploying apache spark," Blog, Jul. 2015, accessed: 03-25-2017. [Online]. Available: https://velvia.github.io/Configuring-Deploying-Spark/

[12] Apache Spark Foundation, "Spark standalone mode - spark 2.1.0 documentation," Web Page, accessed: 04-09-2017. [Online]. Available: http://spark.apache.org/docs/latest/spark-standalone.html

[13] C. Ltd., "Mesos slave | juju," Web Page, accessed: 04-09-2017. [Online]. Available: https://jujucharms.com/u/dataart.telco/mesos-slave/

[14] Oracle, "Java se development kit 8 - downloads," Web Page, accessed: 03-25-2017. [Online]. Available: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

[15] Hortonworks, "What is apache spark," Web Page, accessed: 03-25-2017. [Online]. Available: https://hortonworks.com/apache/spark/#section_8

[16] Apache Software Foundation, "Monitoring and instrumentation - spark 2.1.0 documentation," Web Page, accessed: 03-22-2017. [Online]. Available: http://spark.apache.org/docs/latest/monitoring.html

[17] A. Woodie, "Apache spark: 3 real-world use cases," Article, Mar. 2014, accessed: 20-Mar-2017. [Online]. Available: https://www.datanami.com/2014/03/06/apache_spark_3_real-world_use_cases/

[18] P. Kumar, "Apache spark use cases," Article, Sep. 2016, accessed: 20-Mar-2017. [Online]. Available: https://www.linkedin.com/pulse/apache-spark-use-cases-prateek-kumar

[19] mongoDB, "Apache spark use cases," Web Page, accessed: 20-Mar-2017. [Online]. Available: https://www.mongodb.com/scale/apache-spark-use-cases

[20] A. G. Shoro and T. R. Soomro, "Big data analysis: Apache spark perspective," *Global Journal of Computer Science and Technology*, vol. 15, no. 1, p. 9, 2015. [Online]. Available: https://globaljournals.org/GJCST_Volume15/2-Big-Data-Analysis.pdf

[21] Apache Software Foundation, "Spark programming guide - spark 2.1.0 documentation," Web Page, accessed: 02-22-2017. [Online]. Available: http://spark.apache.org/docs/latest/programming-guide.html

[22] Apache Software Foundation, "Apache license and distribution faq," Web Page, 2016, accessed: 24-Mar-2017. [Online]. Available: http://www.apache.org/foundation/license-faq.html

An overview of Apache THRIFT and its architecture

KARTHIK ANBAZHAGAN[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: kartanba@iu.edu

April 30, 2017

Thrift is a software framework developed at Facebook to accelerate the development and implementation of efficient and scalable cross-language development services. Its primary goal is to enable efficient and reliable communication across programming languages by abstracting the portions of each language that tend to require the most customization into a common library that is implemented in each language. This paper summarizes the how Thrift provides flexibility in use by choosing different layers of the architecture separately.

Keywords: Cloud, Apache Thrift, cross-language, I524

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IR-2008/report.pdf

1. INTRODUCTION

Apache Thrift is an Interface Definition Language [1] (IDL) used to define and create services between numerous languages as a Remote Procedure Call (RPC). Thrift's lightweight framework and its support for cross-language communication makes it more robust and efficient compared to other RPC frameworks like SOA [2] (REST/SOAP). It allows you to create services that are usable by numerous languages through a simple and straightforward IDL. Thrift combines a software stack with a code generation engine to build services that works efficiently and seamlessly between $C++$, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C, Cocoa, JavaScript, Node.js, Smalltalk, and OCaml. In addition to interoperability, Thrift can be very efficient because of a serialization mechanism [3] which can save both space and time. In other words, Apache Thrift lets you create a service to send/receive data between two or more softwares that are written in completely different languages/platforms.

Thrift was originally developed at Facebook and is one of the core parts of their infrastructure. The choice of programming language at Facebook [4] was based on what language was best suited for the task at hand. This flexibility resulted in difficulties when these applications needed to call one another and Facebook needed an application that could meet their needs of interoperability, transport efficiency, and simplicity. Out of this need, they developed efficient protocols and a service infrastructure which became Thrift. Facebook decided to make Thrift an Open Source and finally contributed it to Apache Software Foundation (ASF) in April 2007 in order to increase usage and development. Thrift was later released under the Apache 2.0 license.

2. ARCHITECTURE

Figure. 1 Architecture of Apache Thrift shows the architecture of a model for using the Thrift Stack. It is essential to understand every component of the architecture to understand how Apache Thrift works. It includes a complete stack for creating clients and servers. The top portion of the stack is the user generated code from the Thrift Client-Server definition file. The next layer of the framework are the Thrift generate client and processor codes which also comprises of data structures. The next two important layers are the protocol and transport layers which are part of the Thrift run-time libraries. This provides Thrift the freedom to define a service and change the protocol and transport without regenerating any code. Thrift includes a server infrastructure to tie the protocols and transports together. There are blocking, non-blocking, single and multi-threaded servers. The 'Physical' portion of the stack varies from stack to stack based on the language. For example, for Java and Python network I/O, the built-in libraries are leveraged by the Thrift library, while the C++ implementation uses its own custom implementation. Thrift allows users to choose independently between protocol, transport and server. With Thrift being originally developed in C++, Thrift has the greatest variation among these in the C++ implementation [5].

2.1. Transport Layer

The transport layer provides simple freedom for read/write to/from the network. Each language must have a common interface to transport bidirectional raw data. The transport layer describes how the data is transmitted. This layer seperates the underlying transport from the rest of the system, exposing only the following interface: open, close, isOpen, read, write, and flush

There are multiple transports supported by Thrift:

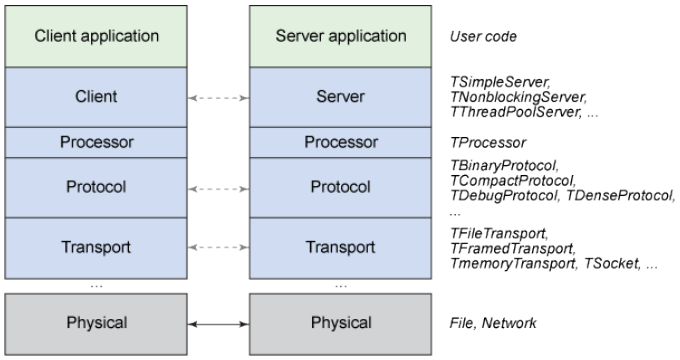


Fig. 1. Architecture of Apache Thrift [6]

1. **TSocket**: The TSocket class is implemented across all target languages. It provides a common, simple interface to a TCP/IP stream socket and uses blocking socket I/O for transport.

2. **TFramedTransport**: The TFramedTransport class transmits data with frame size headers for chunking optimization or non-blocking operation

3. **TFileTransport**: The TFileTransport is an abstraction of an on-disk file to a data stream. It can be used to write out a set of incoming Thrift requests to a file on disk

4. **TMemoryTransport**: Uses memory for I/O operations. For example, The Java implementation uses a simple ByteArrayOutput stream

5. **TZlibTransport**: Performs compression using zlib. It should be used in conjunction with another transport

2.2. Protocol Layer

The second major abstraction in Thrift is the separation of data structure from transport representation. While transporting the data, Thrift enforces a certain messaging structure. That is, it does not matter what method the data encoding is in, as long as the data supports a fixed set of operations that allows it to be read and written by generated code. The Thrift Protocol interface is very straightforward, it supports two things: bidirectional sequenced messaging, and encoding of base types, containers, and structs.

Thrift supports both text and binary protocols. The binary protocols almost always outperforms text protocols, but sometimes text protocols may prove to be useful in cases of debugging. The Protocols available for the majority of the Thrift-supported languages are:

1. **TBinaryProtocol**: A straightforward binary format encoding takes numeric values as binary, rather than converting to text

2. **TCompactProtocol**: Very efficient and dense encoding of data. This protocol writes numeric tags for each piece of data. The recipient is expected to properly match these tags with the data

3. **TDenseProtocol**: It's similar to TCompactProtocol but strips off the meta information from what is transmitted and adds it back at the receiver side

4. **TJSONProtocol**: Uses JSON for data encoding

5. **TSimpleJSONProtocol**: A write-only protocol using JSON. Suitable for parsing by scripting languages.

6. **TDebugProtocol**: Sends data in the form of human-readable text format. It can be well used in debugging applications involving Thrift.

2.3. Processor Layer

A processor encapsulates the ability to read data from input streams and write to output streams. The processor layer is the simplest layer. The input and output streams are represented by protocol objects. Service-specific processor implementations are generated by the Thrift compiler and these generated codes make the Process Layer of the architecture stack. The processor essentially reads data from the wire (using the input protocol), delegates processing to the handler (implemented by the user), and writes the response over the wire (using the output protocol).

2.4. Server Layer

A server pulls together all the various functionalities to complete the Thrift server layer. First, it creates a transport, then specifies input/output protocols for the transport. It then creates a processor based on the I/O protocols and waits for incoming connections. When a connection is made, it hands them off to the processor to handle the processing. Thrift provides a number of servers:

1. **TSimpleServer**: A single-threaded server using standard blocking I/O socket. Mainly used for testing purposes

2. **TThreadPoolServer**: A multi-threaded server with N worker threads using standard blocking I/O. It generally creates five minimum threads in the pool if not specified otherwise

3. **TNonBlockingServer**: A multi-threaded server using non-blocking I/O

4. **THttpServer**: A HTTP server (for JS clients)

5. **TForkingServer**: Forks a process for each request to server

3. ADVANTAGES AND LIMITATIONS OF THRIFT

A few reasons where Thrift is robust and efficient compared to other RPC frameworks are that Thrift leverages the cross-language serialization with lower overhead than alternatives such as SOAP due to use of binary format. Since Thrift generates the client and server code completely [7], it leaves the user with the only task of writing the handlers and invoking the client. Everything including the parameters and returns are automatically validated and analysed. Thrift is more compact than HTTP and can easily be extended to support things like encryption, compression, non blocking IO, etc. Since Protocol Buffers [8] are implemented in a variety of languages, they make interoperability between multiple programming languages simpler.

While there numerous advantages of Thrift over other RPC frameworks, there are a few limitations. Thrift [9] is limited to only one service per server. There can be no cyclic structs. Structs can only contain structs that have been declared before it. Also, a struct also cannot contain itself. Important OOP concepts like inheritance and polymorphism are not supported

and neither can Null be returned by a server. Instead a wrapper struct or value is expected. No out-of-the-box authentication service available between server and client and no Bi-Directional messaging is available in Thrift.

4. CONCLUSION

Thrift provides flexibility in use by choosing different layers of the architecture separately. As mentioned in the advantage section, Thrift usage of the cross-language serialization with lower overheads makes it more efficient compared to other similar technologies. Thirft avoids duplicated work by writing buffering and I/O logic in one place. Thrift has enabled Facebook to build scalable back-end services efficiently. It has been employed in a wide variety of applications at Facebook, including search, logging, mobile, ads, and the developer platform. Application developers can focus on application code without worrying about the sockets layer.

REFERENCES

[1] Apache, "Thrift interface description language," Web Page, 2016. [Online]. Available: https://thrift.apache.org/docs/idl

[2] K. Sandoval, "Microservice showdown – rest vs soap vs apache thrift," May 2015, accessed: 2015-05-19. [Online]. Available: http://nordicapis.com/microservice-showdown-rest-vs-soap-vs-apache-thrift-and-why-it-matters/

[3] P. Kloe, "Benchmarks serializers," Web Page, July 2016, accessed: 2016-07-09. [Online]. Available: https://github.com/eishay/jvm-serializers/wiki

[4] M. Slee, A. Agarwal, and M. Kwiatkowski, "Thrift: Scalable cross-language services implementation," 2013. [Online]. Available: https://thrift.apache.org/static/files/thrift-20070401.pdf

[5] A. Prunicki, "Apache thrift," Web Page, June 2009, accessed: 2009-06-01. [Online]. Available: https://objectcomputing.com/resources/publications/sett/june-2009-apache-thrift/

[6] C. Sun, "Understanding how thrift rpc works," Web Page, March 2015, accessed: 2015-09-22. [Online]. Available: http://sunchao.github.io/posts/2015-09-22-understanding-how-thrift-works.html

[7] S. Dimopoulos, "Generating code with thrift," Web Page, 2013, accessed: 2013-01-01. [Online]. Available: http://thrift-tutorial.readthedocs.io/en/latest/usage-example.html

[8] Google, "Protocol buffers - google's data interchange format," Web Page, 2015, accessed: 2017-03-01. [Online]. Available: https://developers.google.com/protocol-buffers/

[9] C. Maheshwari, "Apache thrift: A much needed tutorial," Web Page, August 2013, accessed: 2013-08-01. [Online]. Available: digital-madness.in/blog/wp-content/uploads/2012/11/BSD_08_2013.8-18.pdf

Hyper-V

ANURAG KUMAR JAIN[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: jainanur@iu.edu

Paper-2, April 30, 2017

A hypervisor or virtual machine monitor (VMM) is computer software, firmware, or hardware, that creates and runs virtual machines. Microsoft Hyper-V Server is the hypervisor-based server virtualization product that allows users to consolidate workloads onto a single physical server [1]. Hyper-V has advantages of being scalable, secure and flexible.

Keywords: cloud, big data, hypervisors, hyper-v, virtualization

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/report.pdf

1. INTRODUCTION

Cloud computing is a booming field and with that, the need for virtualization is also growing. Microsoft has long back stepped into the field of virtualization and is improving in the sector. It brought Microsoft Hyper-V, also codenamed as Viridian and formerly as Windows Server Virtualization to compete with VMware vSphere [2] [3]. It is a native hypervisor which can be used to create virtual machines on x86-64 systems running Windows.

With the release of Windows 8, Hyper-V overtook Windows Virtual-PC as the hardware virtualization component of the client editions of Windows NT. Hyper-V is also available on the Xbox One, in which it would launch both Xbox OS and Windows 10 [2]. Hyper-V supports Windows XP, Vista, Windows 7, Windows 8-8.1, Windows 10, Windows Server 2003-2016, CentOs 5.5-7.0, Red Hat Enterprise Linux 5.5-7.0, Ubuntu 12.04-14.04 among others [3].

2. ARCHITECTURE

Hyper-V maintains isolation of virtual machines in terms of a partition [2]. A partition is a logical unit of isolation in which each guest OS executes. A Hyper-V instance needs to have at least one parent partition, running a supported version of Windows Server (2008 and later). The virtualization stack runs in the parent partition and has direct access to the hardware devices. The child partitions, which host the guest operating systems, are created on parent partitions. A child partition is created by parent partition using the hypercall API, which is the application programming interface exposed by Hyper-V [4].

A child partition does not have direct access to the physical processor. A child partition doesn't even handle its real interrupts. It has a virtual view of the processor and runs in a guest virtual address. Depending on virtual machine configuration, Hyper-V may allow access to a subset of the processors to each partition. The hypervisor handles the interrupts to the processor, and redirects them to the respective partition. Hyper-V can hardware accelerate the address translation of guest virtual address-spaces by using second level address translation provided by the CPU [1].

Direct access to hardware resources is not allowed to the child partitions, but they are allowed have a virtual view of the resources, in terms of virtual devices [4]. Any request to the virtual devices is redirected to the devices in the parent partition, which then manages the requests [4]. The VMBus is a logical channel which enables inter-partition communication. The request and response are redirected via the VMBus [4]. If the devices in the parent partition are also virtual devices, it will be redirected further until it reaches the parent partition, where it will gain access to the physical devices.

3. PREREQUISITES

To install Hyper-V we need to have an x64 based processor with a minimum of 1.4GHz clock speed. We also need to enable hardware-assisted virtualization, this feature is available in processors that include an inbuilt virtualization option specifically, Intel Virtualization Technology or AMD Virtualization. It also requires hardware-enforced Data Execution Prevention (DEP). Specifically, Intel XD bit (execute disable bit) or AMD NX bit (no execute bit)must be enabled [3]. The processor should also support second level address translation. Minimum 2 GB memory with error correcting code or similar technology is required, realistically much more memory is required as each virtual machine requires its own memory. The installation also requires a minimum of 32GB disk space. Apart from the above mentioned requirements, there are other requirements which are not mandatory but required to enable certain features [2].

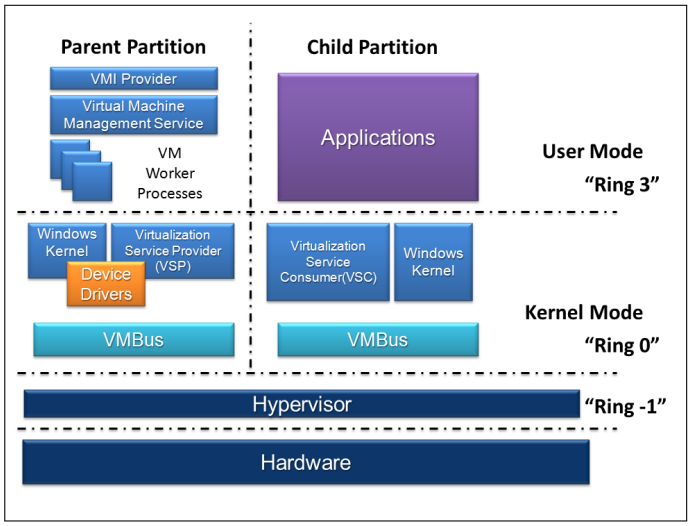


Fig. 1. Hyper-V architechture [2].

4. INSTALLATION

Installation of Hyper-V needs you to install Windows Server 2012 R2, for which you need a bootable device having the same and boot the device using it. Select the operating system you need to install i.e. standard/datacenter. Accept the terms and select install windows and select the drive you want to install the operating system on, we need a minimum of 32GB space for the installation. Set the username and password and then login. After logging on change the host name by going under my computer system properties. Open the server manager and configure according to the requirements which include selecting the roles and feature as required. Select the server name and click next and wait for the installation to complete [3]. Once Hyper-V is installed you can install create virtual networks and create virtual machines on which you can then install the operating system.

5. ADVANTAGES AND FEATURES

With Windows Server 2012, Hyper-V supports network virtualization, multi-tenancy, vhdx disk format supporting virtual hard disks as large as 64TB, offloaded data transfer, cross-premises connectivity and Hyper-V replica. And with Windows Server 2012 R2 also supports shared virtual hard disk, storage quality of services, enhances session mode [2]. Multiple physical servers can be easily consolidated into comparatively fewer servers by implementing virtualization with Hyper-V. Consolidation accommodates the full use of deployed hardware resources. Hyper-V also helps in ease of administration as consolidation and centralization of resources simplifies administration and scale-up or scale-out can be accommodated with much greater ease. With Hyper-V and with virtualization, in general, there are significant cost savings. As separate physical machines are not required for every host and multiple virtual machines can be easily setup on a single physical machine. Hyper-V can be easily managed and a comprehensive Hyper-V management solution is available with System Center Virtual Machine Manager. Additional processing power, network bandwidth, and storage capacity can be accomplished quickly and easily by assigning additional resources from the host computer to the guest virtual machinws [5].

6. LIMITATIONS

Hyper-V does not virtualize audio hardware. It does not support the host/root operating system's optical drives to pass-through guest VMs, as a result, burning to disks are not supported. In Windows Server 2008, Hyper-V does not support live migration of guest VMs where live migration is maintaining network connections and uninterrupted services during VM migration between physical hosts. Although Hyper-V doesn't provide live migration but it tries to eliminate the limitation by having quick migration feature. Also, when Hyper-V is installed it uses VT-x x86 virtualization feature making it unavailable for other solutions due to which software which requires VT-x support can't be installed in parallel [2]. One of the major operating system that is still unsupported includes Fedora 8 and 9 [2].

7. MANAGEMENT

Hyper-V servers can be managed using Windows PowerShell either locally or remotely [6]. By running server manager on a remote computer, a server running in server core mode can be connected. It can also be connected using Microsoft Management Console (MMC) snap-in or by using another computer running Windows, the user can use Remote Desktop Services to run scripts and tools on a server [6]. Server can be switched to graphical user interface mode to use the usual user interface tools to accomplish the tasks and then switch back to server core mode [6].

Hype-V hosts can be managed using Hyper-V manager where the manager lets you manage a small number of Hyper-V hosts, both remote and local. It's gets installed with the installation of Hyper-V Management Tools, which can be installed through a full Hyper-V installation or a tools-only installation [6].

8. COMPARISON BETWEEN HYPER-V AND VSPHERE

Windows Server 2012 R2, VMware vSphere Hypervisor and VMware vSphere 5.5 Enterprise Plus all support 320 logical processors, 4TB of physical memory, 1TB of memory per virtual machine. While Hyper-V and vSphere Enterprise edition both support 64 virtual CPUs per virtual machine, vSphere Hypervisor only support 8. In Hyper-V there can be 1,024 active VMs per host while this is limited to 512 in vSphere. It is interesting to know that Hyper-V supports up to 64 nodes and up to 8,000 virtual machines per in a cluster while vShere Enterprise plus supports 32 and 4,000 respectively [1].

Both VMware vSphere Hypervisor and Hyper-V are free standalone hypervisors, however enterprise edition of vSphere is not. The above information shows that Hyper-V has a number of advantages from a scalability perspective, especially when it comes to comparison with the vSphere Hypervisor [1]. VMware vSphere 5.5 brought a number of scalability increases for vSphere environments, doubling the number of host logical processors supported from 160 to 320, and doubling the host physical memory from 2TB to 4TB, but this still only brings vSphere up to the level that Hyper- V has been offering since September 2012 [1]. Hyper-V also supports double the number of active virtual machines per host, than both the vSphere Hypervisor and vSphere 5.5 Enterprise Plus.

9. CONCLUSION

Hyper-V is a powerful hypervisor introduced by Microsoft with features such as high availability, scalability, reliability, flexibility.

It also supports resource monitoring that helps user track historical data on the use of virtual machines and gain insight into the resource use of specific servers. Hyper-V sees competition from many other supervisors such as vSphere, Qemu, KVM, VirtualBox and provides a tough competition. Hyper-V requires hardware assisted virtualization support from processors and it can only be used with x86-64 processors. In spite of it's limitations it's a popular choice because of the features it provides. The customization options available in Hyper-V provides the user with lot of options to manage the virtual machines as he would like. Hyper-V and Hyper-V hosts can be easily managed using Windows PowerShell and Hyper-V manager tool locally or remotely.

REFERENCES

[1] Microsoft, "Why Hyper-V?" Web Page, Mar. 2016, accessed: 2017-03-22. [Online]. Available: https://download.microsoft.com/download/E/8/E/E8ECBD78-F07A-4A6F-9401-AA1760ED6985/Competitive-Advantages-of-Windows-Server-Hyper-V-over-VMware-vSphere.pdf

[2] Wikipedia, "Hyper-V," Web Page, Mar. 2016, accessed: 2017-03-21. [Online]. Available: https://en.wikipedia.org/wiki/Hyper-V

[3] Microsoft, "Hyper-V Configuration Guide," Web Page, Mar. 2017, accessed: 2017-03-20. [Online]. Available: https://gallery.technet.microsoft.com/Hyper-v-Step-by-step-84632942/file/122329/1/Hyper-vConfigGuide.pdf

[4] Microsoft, "Hyper-V Architecture," Web Page, Mar. 2016, accessed: 2017-03-21. [Online]. Available: https://msdn.microsoft.com/en-us/library/cc768520(v=bts.10).aspx

[5] Microsoft, "Hyper-V Feature Overview," Web Page, Mar. 2017, accessed: 2017-03-24. [Online]. Available: https://msdn.microsoft.com/en-us/library/cc768521(v=bts.10).aspx

[6] Microsoft, "Remotely manage Hyper-V hosts with Hyper-V Manager," Web Page, Mar. 2017, accessed: 2017-03-23. [Online]. Available: https://technet.microsoft.com/en-us/windows-server-docs/compute/hyper-v/manage/remotely-manage-hyper-v-hosts

Retainable Evaluator Execution Framework

PRATIK JAIN

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

Corresponding authors: jainps@iu.edu

Paper-2, April 30, 2017

Apache REEF is a Big Data system that makes it easy to implement scalable, fault-tolerant runtime environments for a range of data processing models on top of resource managers such as Apache YARN and Mesos. The key features and abstractions of REEF are discussed. Two libraries of independent value are introduced. Wake is an event-based-programming framework and Tang is a dependency injection framework designed specifically for configuring distributed systems.

Keywords: REEF, Tang, Wake

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2012/report.pdf

1. INTRODUCTION

With the continuous growth of Hadoop[1] the range of computational primitives expected by its users has also broadened. A number of performance and workload studies have shown that Hadoop MapReduce is a poor fit for iterative computations, such as machine learning and graph processing. Also, for extremely small computations like ad-hoc queries that compose the vast majority of jobs on production clusters, Hadoop MapReduce is not a good fit. Hadoop 2 addresses this problem by factoring MapReduce into two components: an application master that schedules computations for a single job at a time, and YARN, a cluster resource manager that coordinates between multiple jobs and tenants. In spite of the fact that this resource manager, YARN, allows a wide range of computational frameworks to coexist in one cluster, many challenges remain [2].

From the perspective of the scheduler, a number of issues arise that must be appropriately handled in order to scale-out to massive datasets. First, each map task should be scheduled close to where the input block resides, ideally on the same machine or rack. Second, failures can occur at the task level at any step, requiring backup tasks to be scheduled or the job being aborted. Third, performance bottlenecks can cause an imbalance in the task-level progress. The scheduler must react to these stragglers by scheduling clones and incorporating the logical task that crosses the finish line first.

Apache REEF (Retainable Evaluator Execution Framework), a library for developing portable applications for cluster resource managers such as Apache Hadoop YARN or Apache Mesos, addresses these challenges. It provides a reusable control-plane for scheduling and coordinating task-level work on cluster resource managers. The REEF design enables sophisticated optimizations, such as container re-use and data caching, and facilitates workflows that span multiple frameworks. Examples include pipelining data between different operators in a relational system, retaining state across iterations in iterative or recursive data flow, and passing the result of a MapReduce job to a Machine Learning computation.

2. FEATURES

Due to its following critical features, Apache REEF drastically simplifies development of resource managers [3].

2.1. Centralized Control Flow

Apache REEF turns the chaos of a distributed application into various events in a single machine. These events include container allocation, Task launch, completion, and failure.

2.2. Task runtime

Apache REEF provides a Task runtime which is instantiated in every container of a REEF application and can keep data in memory in between Tasks. This enables efficient pipelines on REEF.

2.3. Support for multiple resource managers

Apache REEF applications are portable to any supported resource manager with minimal effort. In addition to this, new resource managers are easy to support in REEF.

2.4. .NET and Java API

Apache REEF is the only API to write YARN or Mesos applications in .NET. Additionally, a single REEF application is free to mix and match tasks written for .NET or Java.

2.5. Plugins

Apache REEF allows for plugins to augment its feature set without hindering the core. REEF includes many plugins, such as a name-based communications between Tasks, MPI-inspired group communications, and data ingress.

As a result of such features Apache REEF shows properties like retainability of hardware resources across tasks and jobs, composability of operators written for multiple computational frameworks and storage backends, cost modeling for data movement and single machine parallelism, fault handling [4] and elasticity.

3. KEY ABSTRACTIONS

REEF is structured around the following key abstractions [5]:

Driver: This is a user-supplied control logic that implements the resource allocation and Task scheduling logic. There is exactly one Driver for each Job. The duration and characteristics of the Job are determined by this module.

Task: This encapsulates the task-level client code to be executed in an Evaluator.

Evaluator: This is a runtime environment on a container that can retain state within Contexts and execute Tasks (one at a time). A single evaluator may run many activities throughout its lifetime. This enables sharing among Activities and reduces scheduling costs.

Context: It is a state management environment within an Evaluator that is accessible to any Task hosted on that Evaluator.

Services: Objects and daemon threads that are retained across Tasks that run within an Evaluator [2]. Examples include caches of parsed data, intermediate state, and network connection pools.

4. WAKE AND TANG

The lower levels of REEF can be decoupled from the data models and semantics of systems built atop it. This results in two standalone systems, Tang and Wake which are both language independent and allow REEF to bridge the JVM and .NET.

Tang is a configuration management and checking framework [6]. It emphasizes explicit documentation and ability of configurations and applications of being automatically checkable instead of ad-hoc, application-specific configuration and bootstrapping logic. It not only supports distributed, multi-language applications but also gracefully handles simpler use cases. It makes use of dependency injection to automatically instantiate applications. Given a request for some type of object, and information that explains how dependencies between objects should be resolved, dependency injectors automatically instantiate the requested object and all of the objects it depends upon. Tang makes use of a few simple wire formats to support remote and even cross-language dependency injection.

Wake is an event-driven framework based on ideas from SEDA, Click, Akka and Rx [7]. It is general purpose in the sense that it is designed to support computationally intensive applications as well as high-performance networking, storage, and legacy I/O systems. Wake is implemented to support high-performance, scalable analytical processing systems i.e. big data applications. It can be used to achieve high fanout and low latency as well as high-throughput processing and it can thus aid to implement control plane logic and the data plane.

Wake is designed to work with Tang. This makes it extremely easy to wire up complicated graphs of event handling logic. In addition to making it easy to build up event-driven applications, Tang provides a range of static analysis tools and provides a simple aspect-style programming facility that supports Wake's latency and throughput profilers.

5. RELATIONSHIPS WITH OTHER APACHE PRODUCTS

Given REEF's position in the big data stack, there are three relationships to consider: Projects that fit below, on top of, or alongside REEF in the stack.

5.1. Below REEF

REEF is designed to facilitate application development on top of resource managers like Mesos and YARN. Hence, its relationship with the resource managers is symbiotic by design.

5.2. On Top of REEF

Apache Spark, Giraph, MapReduce and Flink are some of the projects that logically belong at a higher layer of the big data stack than REEF. Each of these had to individually solve some of the issues REEF addresses.

5.3. Alongside REEF

Apache builds library layers on top of a resource management platform. Twill, Slider, and Tez are notable examples in the incubator [8]. These projects share many objectives with REEF. Twill simplifies programming by exposing a programming model. Apache Slider is a framework to make it easy to deploy and manage long-running static applications in a YARN cluster. Apache Tez is a project to develop a generic Directed Acyclic Graph (DAG) processing framework with a reusable set of data processing primitives. Apache Helix automates application-wide management operations which require global knowledge and coordination, such as repartitioning of resources and scheduling of maintenance tasks. Helix separates global coordination concerns from the functional tasks of the application with a state machine abstraction.

6. CONCLUSION

REEF is a flexible framework for developing distributed applications on resource manager services. It is a standard library of reusable system components that can be easily composed into application logic. It possesses properties of retainability, composability, cost modeling, fault handling and elasticity. Its key components are driver, task, evaluator, context and services. Its relationship with projects above it, below it and alongside it in the big data stack is discussed. Thus, a brief overview of REEF is shown here.

ACKNOWLEDGEMENTS

The author thanks Prof. Gregor von Laszewski and all the course AIs for their continuous technical support.

REFERENCES

[1] The Apache Software Foundation, "Welcome to apache™ hadoop®!" Web Page, Mar. 2017, accessed 2017-03-28. [Online]. Available: http://hadoop.apache.org/

[2] Byung-Gon Chun, Chris Douglas, Shravan Narayanamurthy, Josh Rosen, Tyson Condie, Sergiy Matusevych, Raghu Ramakrishnan, Russell Sears, Carlo Curino, Brandon Myers, Sriram Rao, Markus Weimer, "Reef: Retainable evaluator execution framework," in *VLDB Endowment, Vol. 6, No. 12*, Aug. 2013. [Online]. Available: http://db.disi.unitn.eu/pages/VLDBProgram/pdf/demo/p841-sears.pdf

[3] The Apache Software Foundation, "Apache reef™ - a stdlib for big data," Web Page, Nov. 2016, accessed 2017-03-15. [Online]. Available: http://reef.apache.org/

[4] Techopedia Inc., "Retainable evaluator execution framework (reef)," Web Page, Jan. 2017, accessed 2017-03-17. [Online]. Available: https://www.techopedia.com/definition/29891/retainable-evaluator-execution-framework-reef

[5] Markus Weimer, Yingda Chen, Byung-Gon Chun, Tyson Condie, Carlo Curino, Chris Douglas, Yunseong Lee, Tony Majestro, Dahlia Malkhi, Sergiy Matusevych, Brandon Myers, Shravan Narayanamurthy, Raghu Ramakrishnan, Sriram Rao, Russell Sears, Beysim Sezgin, Julia Wang, "Reef: Retainable evaluator execution framework," in *Proc ACM SIGMOD Int Conf Manag Data. Author manuscript*, Jan. 2016, pp. 1343–1355. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4724804/

[6] The Apache Software Foundation, "Tang," Web Page, Nov. 2016, accessed 2017-03-15. [Online]. Available: http://reef.apache.org/tang.html

[7] The Apache Software Foundation, "Wake," Web Page, Dec. 2016, accessed 2017-03-15. [Online]. Available: http://reef.apache.org/wake.html

[8] The Apache Software Foundation, "Reefproposal - incubator," Web Page, Aug. 2014, accessed 2017-03-15. [Online]. Available: https://wiki.apache.org/incubator/ReefProposal

A brief introduction to OpenCV

SAHITI KORRAPATI[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: sakorrap@iu.edu, S17-IR-2013*

techpaper-2, May 7, 2017

This paper provides a brief introduction to OpenCV. OpenCV is an open source computer vision and machine learning software library, which was originally introduced more than a decade ago by Intel. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms [1].

Keywords: OpenCV, Computer Vision, Machine Learning

https://github.com/sakorrap/sp17-i524/blob/master/paper2/S17-IR-2013/report.pdf

INTRODUCTION

Computer Vision is the science of programming for a computer to process and understand images and videos so that the machines can detect and recognize faces, identify objects, classify human actions in videos, track moving objects, etc [2]. In order to advance vision research and disseminate vision knowledge, it is highly critical to have a library of programming functions with the optimized and portable code [3]. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception [2].
In the early days of OpenCV, the goals of the project were described as [4]:

1. Advance vision research by providing not only open but also optimized code for basic vision infrastructure. No more reinventing the wheel.

2. Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable.

3. Advance vision-based commercial applications by making portable, performance-optimized code available for free with a license that did not require code to be open or free itself.

It is an open-source BSD-licensed library that now includes several hundreds of computer vision and machine learning algorithms. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. Since the official launch of OpenCV in 1999, a number of programmers have contributed to the most recent library developments. It has C++, C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. Though machine learning algorithms are added to OpenCV to support computer vision develpment, they can be used in other applications like speech recognition and anomaly detection. CUDA and OpenCL interfaces are being actively developed right now [2].

PLATFORMS FOR OPENCV

OpenCV was designed to be a cross-platform tool, due to which it is completely written in C. This makes it portable to any commercial system, right from PCs, Macs, to robotic on board computers as the C compilers were stable during the initial days of development. Moreover, low-level optimization is possible through C. When it comes to Computer Vision, such optimizations may easily lead to significant speedup [5]. Since OpenCV 2.0 version, there is a C and C++ interface also, and all new packages are written in C++. However, to encourage widespread use, wrappers for popular programming languages like Python and Java have been developed and OpenCV can be used both on mobile and desktop. We look at the latest additions to OpenCV platforms [5]:

CUDA

CUDA is a parallel computing platform and application programming interface (API) model created by Nvidia. It allows software developers and software engineers to use a CUDA-enabled graphics processing unit (GPU) for general purpose processing [6].

In 2010 a new module that provides GPU acceleration was added to OpenCV. The 'gpu' module covers a significant part of the library's functionality and is still in active development. It is implemented using CUDA and therefore benefits from the CUDA ecosystem, including libraries such as NPP (NVIDIA Performance Primitives). With the addition of CUDA acceleration to OpenCV, developers can run more accurate and sophisticated OpenCV algorithms in real-time on higher-resolution images while consuming less power.

Android

Since 2010 OpenCV was ported to the Android environment, it allows to use the library in mobile applications development.

iOS

In 2012 OpenCV development team actively worked on adding extended support for iOS. Full integration is available since version 2.4.2 (2012).

OpenCL

In 2011 a new module providing OpenCL accelerations of OpenCV algorithms was added to the library. This enabled OpenCV-based code taking advantage of heterogeneous hardware, in particular utilize potential of discrete and integrated GPUs. Since version 2.4.6 (2013) the official OpenCV WinMega-Pack includes the OpenCL module.

In the 2.4 branch OpenCL-accelerated versions of functions and classes were located in a separate ocl module and in a separate namespace (*cv::ocl*), and often had different names (e.g. *cv::resize()* vs *cv::ocl::resize()* and *cv::CascadeClassifier* vs *cv::ocl::OclCascadeClassifier*) that required a separate code branch in user application code. Since OpenCV 3.0 (master branch as of 2013) the OpenCL accelerated branches transparently added to the original API functions and are used automatically when possible/sensible.

MODULES IN OPENCV

OpenCV was built as a modular program, which means there are several shared or static libraries to pick from. An overview of the modules present in the latest version of OpenCV (3.2.0) [7]:

1. **Core functionality** a compact module defining basic data structures, including the dense multi-dimensional array Mat and basic functions used by all other modules.
2. **Image processing** an image processing module that includes linear and non-linear image filtering, geometrical image transformations (resize, affine and perspective warping, generic table-based remapping), color space conversion, histograms, and so on.
3. **video** a video analysis module that includes motion estimation, background subtraction, and object tracking algorithms.
4. **calib3d** basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo correspondence algorithms, and elements of 3D reconstruction.
5. **features2d** salient feature detectors, descriptors, and descriptor matchers.
6. **objdetect** detection of objects and instances of the predefined classes (for example, faces, eyes, mugs, people, cars, and so on).
7. **highgui** an easy-to-use interface to simple UI capabilities.
8. **Video I/O** an easy-to-use interface to video capturing and video codecs.
9. **gpu** GPU-accelerated algorithms from different OpenCV modules.
10. There are other helper modules, such as FLANN and Google test wrappers, Python bindings, and others.

SETUP AND CONFIGURATION

Since, Python is the most popular language for Machine Learning and Computer Vision, set up and configuration of OpenCV for Python on Windows may be done as shown below.

Setting up OpenCV Python on Windows

OpenCV requires numpy and matplotlib (optional but recommended) packages to be installed in a Python 2.7 environment to be able to run successfully. After making sure that the dependencies are installed, the following steps will help to setup OpenCV for Python [8]:

1. Download latest OpenCV release from sourceforge site [9] and double-click to extract it.
2. Go to *opencv/build/python/2.7* folder.
3. Copy *cv2.pyd* to *C:/Python27/lib/site-packages*.

Now OpenCV will work as a Python library named cv2. For checking if it works, type the following code in Python environment.

```
import cv2
```

LICENSING

OpenCV is an open source software, and OpenCV allows redistribution in terms of source or binary forms, with or without modifications. All re distributions should bear the copywright information provided at the OpenCV licensing page [10]. The source code is available on Github [11].

USE CASE

OpenCV library can be used for image recognition technology. There are numerous applications of image recognition, like facial recognition for security purposes, facial tagging in cameras, image and video processing in self driving cars. For instance, take a data set containing pictures taken from a forest surveillance cameras, we can process the images to find out the number of animals at any given point of time. This will be helpful in keeping track of endangered animals and to keep a check on animal poaching.
A sample code of loading an image of a watch for the purpose of image recognition using OpenCV in Python is demonstrated below:
Here, we use numpy and matplotlib libraries which we installed earlier [12].

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread('watch.jpg',cv2.IMREAD_GRAYSCALE)
cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Images can also be displayed for which sample code is shown below [12]:

```
plt.imshow(img, cmap = 'gray', interpolation = 'bicubic')
plt.xticks([]), plt.yticks([])  # to hide tick values on X and
plt.plot([200,300,400],[100,200,300],'c', linewidth=5)
plt.show()
```

USEFUL RESOURCES

The OpenCV website has detailed and structured documentation for its modules, for all operating systems and platforms. Further reading is suggested based on the requirements of OS, platform and language by clicking on the version of OpenCV that is currently in use [13].

CONCLUSION

OpenCV is one of the leading Computer Vision software which offers various modules for image and video processing and a library of various Machine learning algorithms. OpenCV is an ever growing platform owing to its open source nature. OpenCV is versatile in terms of its operating systems, platforms and programming languages, which makes it an even more popular tool. The open source Robot Operating System (ROS) uses OpenCV as its primary image processing software. This expands the usage of OpenCv even further.

ACKNOWLEDGEMENTS

The author thanks Professor Gregor Von Laszewski and all the AIs of big data class for the guidance and technical support.

REFERENCES

[1] OpenCV team, "Opencv library," Web Page, 2017, accessed mar-30-2017. [Online]. Available: http://opencv.org/

[2] OpenCV team, "About - opencv library," Web Page, 2017, accessed mar-30-2017. [Online]. Available: http://opencv.org/about.html

[3] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek, "A brief introduction to opencv," in *2012 Proceedings of the 35th International Convention MIPRO*, May 2012, pp. 1725–1730.

[4] Wikipedia, "History," Web Page, 2017, accessed mar-30-2017. [Online]. Available: https://en.wikipedia.org/wiki/OpenCV

[5] OpenCV team, "Platforms - opencv library," Web Page, 2017, accessed mar-30-2017. [Online]. Available: http://opencv.org/platforms/

[6] Wikipedia, "Cuda," Web Page, 2017, accessed mar-30-2017. [Online]. Available: https://en.wikipedia.org/wiki/CUDA

[7] OpenCV team, "Opencv: Introduction," Web Page, 2017, accessed mar-30-2017. [Online]. Available: http://docs.opencv.org/3.2.0/d1/dfb/intro.html

[8] OpenCV team, "Install opencv-python in windows," Web Page, 2017, accessed mar-30-2017. [Online]. Available: http://docs.opencv.org/3.2.0/d5/de5/tutorial_py_setup_in_windows.html

[9] Slashdot Media, "Opencv," Web Page, 2017, accessed mar-30-2017. [Online]. Available: https://sourceforge.net/projects/opencvlibrary/files/

[10] OpenCV team, "License - opencv," Web Page, 2017, accessed mar-30-2017. [Online]. Available: http://opencv.org/license.html

[11] A. Pavlenko, "Opencv," Github Repository, 2017, accessed mar-30-2017. [Online]. Available: https://github.com/opencv

[12] PythonProgramming, "Opencv with python intro and loading images tutorial," Web Page, 2017, accessed apr-16-2017. [Online]. Available: https://pythonprogramming.net/loading-images-python-opencv-tutorial/

[13] OpenCV team, "Opencv documentation index," Web Page, 2017, accessed mar-30-2017. [Online]. Available: http://docs.opencv.org/

AUTHOR BIOGRAPHIES

Sahiti Korrapati is pursuing her MSc in Data Science from Indiana University Bloomington

An Overview of Pivotal Web Services

HARSHIT KRISHNAKUMAR[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: harkrish@iu.edu, S17-IR-2014

Project-02, April 30, 2017

Pivotal Web Services is a platform as a service (PAAS) provider which allows developers to deploy applications written in six programming languages. PWS provides the infrastructure to host applications on the cloud, and allows vertical scaling for each instance and horizontal scaling for the application. PWS is built on CloudFoundry, an open source software for hosting applications on the cloud. This paper presents the different features of Pivotal Web Services and a basic overview of hands-on of application deployment in Pivotal Web Services.

Keywords: Cloud, I524, Web Services

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IR-2014/report.pdf

1. INTRODUCTION

The current scenario for software product based companies is such, that coming up with ground breaking ideas to add extra functionality for an existing application is simply not enough. They need to be able to get it out to the users as quickly as possible, else they loose ground to competitors who might have already implemented it. To make software development and deployment process quicker, software companies follow a few methods and concepts. Pivotal Web Services comes in this line of thought, where it allows the application developer to focus on just the application development and getting the business requirements right, without worrying about platform compatibility, dependencies and differences between production/development/testing environment. PWS is built based on Cloud Foundry which is one of the leading open source PAAS services [1]. In order to comprehend the need for a service like PWS, one would require a basic knowledge of the entire process of agile, devops and PAAS.

1.1. Agile Development

With the widespread use of Internet to push quick updates and the emergence of automation of methods used in software testing and deployment, software companies are moving away from traditional waterfall methodology to agile development practices, which emphasizes on iterative development where there is high collaboration between self organizing and cross functional teams to evolve requirements and solutions. Agile methods encourage deployment of high quality and goal oriented software in quick successions, and any feedback and changes will be handled in the next update version.

1.2. DevOps

Devops is a set of practices for software testing and deployment which enables Agile development. Typically there is latency to for the development process that there are many manual tasks involved. Devops sets standards to automate testing and to ensure that production, testing and development environments are in sync. It gives greater responsibility and access to developers for easy testing and development. With automated processes for testing, developers would get their feedback within minutes, and they can work on fixes. The final aspect of devops is to automate deployment, there are software programs to automatically deploy the software on a host of servers with the right configurations and connections, thus reducing manual effort and latency.

1.3. Platform as a service

The concept of containers started gaining popularity considering the advantages of modularity in software development. Containers in software development serve the purpose of building modular software. A container will have the actual software along with all its dependencies and methods.

Platform as a service (PaaS) or application platform as a service (aPaaS) is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app [2]. PaaS providers generally provide a cloud environment to deploy the application on, the networks, servers, OS, storage, databases and other services to run applications. This removes the hassles of maintaining and running the servers and systems for an application from the developers, and also minimises the risk of server failures.

1.4. Cloud Foundry

Cloud Foundry is an open source PAAS software provider. It provides with all the software and tools required to host applications on multiple clouds. Cloud Foundry does not offer the hardware for hosting clouds, there are many commercial options which provide the platform hardware along with hosted Cloud Foundry software, which takes the responsibility of handling and maintaining the cloud hardware away from application developers.

1.5. Pivotal Web Services

PWS is built based on an open source PaaS Cloud Foundry along with some proprietary additions such as Pivotal's Developer Console, Billing Service and Marketplace [3]. PWS offers hosted cloud systems with a web interface for managing the environment, and a number of pre-provisioned services like relational databases and messaging queues [4]. Pivotal Cloud Foundry enables developers to provision and bind web and mobile apps with platform and data services such as Jenkins, MongoDB, Hadoop, etc. on a unified platform.

2. FEATURES OF PWS

PWS offers many different options to deploy and manage software [5].

2.1. Upload

There is a single command way to upload software developed on local to the cloud. The code is transformed into a running application on the cloud. The steps to follow for uploading an application with name <APP-NAME> is given in [6].

2.2. Stage

Behind the scenes, the deployed application goes through staging scripts called buildpacks to create a ready-to-run package. Buildpacks are software packets that provide framework and runtime support for applications, and they are provided along with PWS cloud. Buildpacks typically examine user-provided artifacts to determine what dependencies to download and how to configure applications to communicate with bound services. Cloud Foundry automatically detects which buildpack is required and installs it on the Diego cell where the application needs to run [7].

For example, if a particular application requires d3.js to run and needs to connect to a database, buildpacks will determine that the application needs these dependencies in order to run and attach d3.js packet with the application and provide connectors to connect to the database.

2.3. Distribute

Deigo is the container management system for Cloud Foundry, which handles application scheduling and management. Each application VM has a Diego Cell that executes application start and stop actions locally, manages the VM's containers, and reports app status and other data [8].

2.4. Run

Applications receive entry in a dynamic routing tier, which load balances traffic across all app instances.

3. LICENSING

Though Cloud Foundry is open source, it is not easy to maintain a cloud and setup the architecture by a developer. PWS is charges for the use of its services, with a monthly cost depending upon the memory of application instance and number of instances.

4. USE CASES

PWS can be used for a range of applications, from running websites to maintaining mobile applications. For example, if we need to host a website which accesses data, we can write the base code and deploy to PWS cloud.

For instance, if there is a Web Page that has to be hosted on cloud, we need to create an account in Pivotal and create the command line interface. Normally, deploying a web page requires web servers like Apache or Nginx, but with Pivotal it will automatically take care of the web server. We need to copy the web page HTML files in our local to the cloud where application needs to be hosted. Next we login to the Pivotal Cloud instance by giving username and password, and create a staticfile. Last step is to push the application.

```
cf login -a https://api.run.pivotal.io
touch Staticfile
cf push <<application file name>>
```

We can verify the deployed webpage using the link which we will get after the above steps.

5. CONCLUSION

PWS is a hosted cloud platform service, which uses Cloud Foundry open source platform. It has options for scaling and updating the cloud with no downtime. As given in Section 2 (Features of PWS) there are a few basic commands to upload an application, and PWS automatically binds applications with dependencies and configurations required. PWS allows developers to concentrate on their business requirements and developing applications, rather than hosting and hardware requirements. PWS also makes up-scaling and downscaling easy. [9].

6. FURTHER EDUCATION

Further learning about Pivotal is encouraged and informative materials can be found at the Pivotal homepage [10].

ACKNOWLEDGEMENTS

The author thanks Professor Gregor Von Lazewski for providing us with the guidance and topics for the paper. The author also thanks the AIs of Big Data Class for providing the technical support.

REFERENCES

[1] Pivotal Software, Inc, "Pivotal web services | home," Web Page, 2017. [Online]. Available: https://run.pivotal.io/

[2] Wikipedia, "Platform as a service," Web Page, Mar. 2017. [Online]. Available: https://en.wikipedia.org/wiki/Platform_as_a_service

[3] J. Clark, "Pivotal fluffs up *sigh* cloud foundry *sigh* cloud for battle in the *sigh* cloud," Web Page, May 2014. [Online]. Available: https://www.theregister.co.uk/2014/05/08/pivotal_web_services_launch/

[4] C. Page, "Difference between cloud foundry and pivotal web services," Web Page, Jun. 2015. [Online]. Available: http://stackoverflow.com/a/30899157

[5] Pivotal Software, Inc., "Pivotal web services | features," Web Page, 2017. [Online]. Available: https://run.pivotal.io/features/
[6] Pivotal Software, Inc., "Deploy an application," Web Page, 2017. [Online]. Available: http://docs.run.pivotal.io/devguide/deploy-apps/deploy-app.html#push
[7] Pivotal Software, Inc., "Buildpacks," Web Page, 2017. [Online]. Available: http://docs.run.pivotal.io/buildpacks/
[8] Pivotal Software, Inc., "Diego architecture," Web Page, 2017. [Online]. Available: https://docs.run.pivotal.io/concepts/diego/diego-architecture.html
[9] Pivotal Software, Inc., "Pivotal cloud foundry: The leading enterprise platform powered by cloud foundry," Web Page, 2017. [Online]. Available: https://content.pivotal.io/datasheets/pivotal-cloud-foundry-the-leading-enterprise-platform-powered-by-cloud-foundry
[10] Pivotal Software, Inc., "Agile | pivotal," Web Page, 2017. [Online]. Available: https://pivotal.io/agile

AUTHOR BIOGRAPHIES

Harshit Krishnakumar is pursuing his MSc in Data Science from Indiana University Bloomington

An Overview of Apache Avro

ANVESH NAYAN LINGAMPALLI[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: anveling@umail.iu.edu

April 30, 2017

Apache Avro is a data serialization system, which uses JSON based schemas and RPC calls to send the data. Hadoop based tools natively support Avro for serialization and deserialization of the data.

Keywords: apache, avro, serialization

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2016/report.pdf

1. INTRODUCTION

Apache Avro [1] is a data serialization system. Data serialization is a mechanism to translate data in a computer environment, such as memory buffer, data structures, object state into binary or textual form. Java and Hadoop provides serialization APIs, which are java based. Apache Avro is a language independent system, that can be processed by multiple languages [2].

Avro is heavily dependent on schemas. These schemas are defined with JSON that simplifies its implementations with its libraries. Different schemas can be used for serialization and deserialization, and Avro will handle the missing fields or extra fields [3]. When Avro data is stored in a file, its schema is also stored with it. Avro stores the data definition in JSON format, which makes it easier to interpret.

Apache Avro provides rich data structures, compact binary data format, a container file, facility for remote procedure calls (RPC) and integration with dynamic languages [1]. In Remote procedure calls, the client and server exchange schemas during the connection. This exchange helps when there are missing fields or extra fields. Avro works well with Hadoop MapReduce, which provides the large scale processing across many processors to do the calculations simultaneously and efficiently.

2. COMPONENTS OF AVRO

Avro has two main components, data serialization and remote procedure call (RPC) support.[4]

2.1. Data Serialization

Java provides a mechanism, called object serialization where an object can be represented as a sequence of bytes that includes the object's data as well as information about the object's type and the types of data stored in the object. To serialize data using Avro, 1) define an Avro Schema. 2) compile the schema using Avro utility. 3) Java code corresponding to that schema is obtained. 4) populate the schema with data. 5) serialize the data using Avro library[**?**].

2.2. Remote Procedure Call

In a Remote Procedure Call, the client and server exchange schemas in the connection handshake. (This can be optimized so that, for most calls, no schemas are actually transmitted.) Since both client and server both have the other's full schema, correspondence between same named fields, missing fields, extra fields, etc. can all be easily resolved[5].

3. SPECIFICATIONS

3.1. Encoding

Avro specifies two serialization encodings, binary and JSON. Binary encoding is smaller and faster, but not efficient in the case of debugging and web-based applications, where JSON encoding is appropriate[6].

3.2. Object Container Files

Avro includes an object container file format. Objects stored in blocks that may be compressed and synchronization markers are used to permit splitting of the files for MapReduce processing. File consists of, a header followed by one or more data blocks. Header consists of four bytes 'O', 'b', 'j', 1. It also consists of the file metadata, including the schema and a 16-byte, randomly generated sync marker[7]. Currently used file metadata properties are, avro.schema which contains the schema of the objects stored in file avro.codec contains the name of the compression codec used to compress the blocks.

A file data block consists of, count of the objects in the block, size of bytes of the serialized objects in current block, the serialized objects and a 16-byte sync marker[7].

Each block's binary data can be efficiently extracted without deserializing the contents. The combination of the block size, object counts, and sync markers enable detection of corrupt blocks.

4. KEY FEATURES OF APACHE AVRO

Apache Thrift and Google's Protocol buffers are the competent libraries with Apache Avro. But Avro is fundamentally different from these frameworks. The key features of the Apache Avro are, Dynamic typing, untagged data, no manually-assigned field IDs.

4.1. Dynamic Typing

Serialization and deserialization without the code generation is possible in Apache Avro. Data is always accompanied by a schema that permits full processing of the data. This feature of the Avro, makes it possible to construct data-processing systems and languages. Avro creates binary structure format that is both compressible and splittable [3].

4.2. Untagged Data

Binary data with a schema together, allows the data to be written without any overhead. This feature provides faster data processing and compact data encoding.

4.3. Schema Evolution

Avro cleanly handles schema changes such as missing fields, added fields and changed fields. By using this feature, new programs can read old data and old programs can read new data.

5. APPLICATION

Primary use of Apache Avro is in Apache Hadoop where it can provide both serialization format for persistent data, and a wire format for communication between Hadoop nodes.

Avro was designed for Hadoop for making it interoperable across different languages. With Avro serialization, Pig utilizes AvroStorage().

5.1. Using Avro with Eventlet

Eventlet[4] is a concurrent networking library for python that allows the changing of the code, to run the code, instead of writing it. RPC support from the Avro combined with Eventlet is used for building highly concurrent network-based services. Avro is present in the transport layer on top of HTTP for RPC calls. It POSTS binary data to the server and processes the response. Eventlet.wsgi (Web server Gateway Interface) is used to build RPC server.

6. DISADVANTAGES

Avro is not the fastest in serialization process, but it is one of the faster frameworks. The syntax of Avro can be error prone. And error handling is complex when compared with Protocol buffers and Thrift.

7. EDUCATIONAL MATERIAL

Tutorialspoint.com [3]for Apache Avro provides the resources to use Avro for deserialization and serialization of the data. Apache Avro's [1] website provides documentation for understanding, deploying and managing the framework.

8. CONCULSION

Apache Avro is a framework, which allows the serialization of data that has schema built in it. The serialization of data is results in compact binary format, which does require proxy objects. Instead of using generated proxy object libraries and strong typing, Avro heavily relies on the schema that are sent along with serialized data.

REFERENCES

[1] "Apache avro," Web Page, accessed: 2017-03-21. [Online]. Available: https://avro.apache.org/

[2] "Apache Avro documentation," Web Page, accessed: 2017-03-21. [Online]. Available: https://avro.apache.org/docs/1.7.7/index.html

[3] "Avro tutorial," Web Page, accessed: 2017-03-22. [Online]. Available: https://www.tutorialspoint.com/avro/avro_overview.html

[4] "Using Avro with Eventlet," Web Page, accessed: 2017-03-22. [Online]. Available: http://unethicalblogger.com/2010/05/07/how-to-using-avro-with-eventlet.html

[5] "Rpc by avro," Web Page, accessed: 2017-03-22. [Online]. Available: https://www.igvita.com/2010/02/16/data-serialization-rpc-with-avro-ruby/

[6] "Encoding in avro," Web Page, accessed: 2017-03-22. [Online]. Available: https://avro.apache.org/docs/1.8.1/spec.html

[7] "Apache avro wiki," Web Page, accessed: 2017-03-22. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Avro

An Overview of Pivotal HD/HAWQ and its Applications

VEERA MARNI[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: vmarni@umail.iu.edu*

April 30, 2017

Pivotal HDB is the Apache Hadoop native SQL database powered by Apache HAWQ for data science and machine learning workloads. It can be used to gain deeper and actionable insights into data with out the need from moving data to another platform to perfrom advanced analytics.

Keywords: pivotal, hd, hawq, data science

https://github.com/narayana1043/sp17-i524/blob/master/paper2/S17-IR-2017/report.pdf

1. INTRODUCTION

Pivotal-HAWQ[1] is a Hadoop native SQL query engine that combines the key technological advantages of MPP database with the scalability and convenience of Hadoop. HAWQ reads data from and writes data to HDFS natively. HAWQ delivers industry-leading performance and linear scalability. It provides users the tools to confidently and successfully interact with petabyte range data sets. HAWQ provides users with a complete, standards compliant SQL interface.

An MPP database is a database that is optimized to be processed in parallel for many operations to be performed by many processing units at a time.[2] HAWQ breaks complex queries into small tasks and distributes them to MPP query processing units for execution. HAWQ's basic unit of parallelism is the segment instance. Multiple segment instances on commodity servers work together to form a single parallel query processing system. A query submitted to HAWQ is optimized, broken into smaller components, and dispatched to segments that work together to deliver a single result set. All relational operations - such as table scans, joins, aggregations, and sorts - simultaneously execute in parallel across the segments. Data from upstream components in the dynamic pipeline are transmitted to downstream components through the scalable User Datagram Protocol (UDP)[3] interconnect.

Based on Hadoop's distributed storage, HAWQ has no single point of failure and supports fully-automatic online recovery. System states are continuously monitored, therefore if a segment fails, it is automatically removed from the cluster. During this process, the system continues serving customer queries, and the segments can be added back to the system when necessary.

2. ARCHITECTURE OF HAWQ

In a typical HAWQ deployment, each slave node has one physical HAWQ segment, an HDFS DataNode and a NodeManager[4] installed. Masters for HAWQ, HDFS and YARN are hosted on separate nodes.[5]. HAWQ is tightly integrated with YARN, the Hadoop resource management framework, for query resource management. HAWQ caches containers from YARN in a resource pool and then manages those resources locally by leveraging HAWQ's own finer-grained resource management for users and groups.

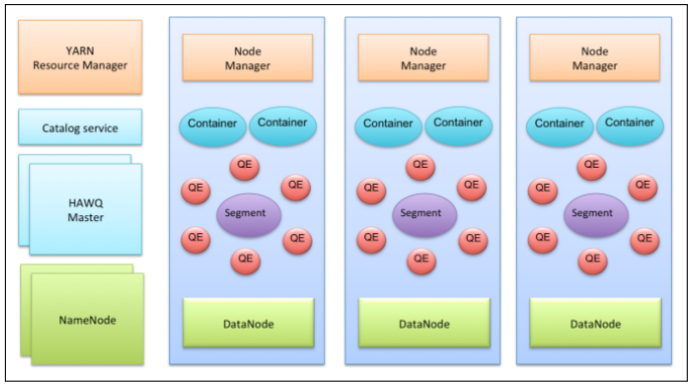


Fig. 1. The following diagram provides a high-level architectural view of a typical HAWQ deployment.[6].

2.1. HAWQ Master

The HAWQ master is the entry point to the system. It is the database process that accepts client connections and processes the SQL commands issued. The HAWQ master parses queries, optimizes queries, dispatches queries to segments and coordinates the query execution. End-users interact with HAWQ through the master and can connect to the database using client programs such as psql or application programming interfaces (APIs) such as JDBC[7] or ODBC[8].The master is where the

global system catalog resides. The global system catalog is the set of system tables that contain metadata about the HAWQ system itself. The master does not contain any user data; data resides only on HDFS.

2.2. HAWQ Segment

In HAWQ, the segments are the units that process data simultaneously. There is only one physical segment on each host. Each segment can start many Query Executors (QEs) for each query slice. This makes a single segment act like multiple virtual segments, which enables HAWQ to better utilize all available resources.

2.3. HAWQ Interconnect

The interconnect is the networking layer of HAWQ. When a user connects to a database and issues a query, processes are created on each segment to handle the query. The interconnect refers to the inter-process communication between the segments, as well as the network infrastructure on which this communication relies.

2.4. HAWQ Resource Manager

The HAWQ resource manager obtains resources from YARN and responds to resource requests. Resources are buffered by the HAWQ resource manager to support low latency queries. The HAWQ resource manager can also run in standalone mode. In these deployments, HAWQ manages resources by itself without YARN.

2.5. HAWQ Catalog Service

The HAWQ catalog service stores all metadata, such as UDF/UDT[9] information, relation information, security information and data file locations.

2.6. HAWQ Fault Tolerance Service

The HAWQ fault tolerance service (FTS) is responsible for detecting segment failures and accepting heartbeats from segments.

2.7. HAWQ Dispatcher

The HAWQ dispatcher dispatches query plans to a selected subset of segments and coordinates the execution of the query. The dispatcher and the HAWQ resource manager are the main components responsible for the dynamic scheduling of queries and the resources required to execute them.

3. KEY FEATURES OF PIVOTAL HDB/HAWQ

3.1. High-Performance Architecture

Pivotal HDB's parallel processing architecture delivers high performance throughput and low latency (potentially, near-real-time) query responses that can scale to petabyte-sized datasets. Pivotal HDB also features a cutting-edge, cost-based SQL query optimizer and dynamic pipelining technology for efficient performance operation.

3.2. Robust ANSI SQL Compliance

Pivotal HDB complies with ANSI SQL-92, -99, and -2003 standards, plus OLAP[10] extensions. Leverage existing SQL expertise and existing SQL-based applications and BI/data visualization tools. Execute complex queries and joins, including roll-ups and nested queries.

3.3. Deep Analytics and Machine Learning

Pivotal HDB integrates statistical and machine learning capabilities that can be natively invoked from SQL and applied natively to large data sets across a Hadoop cluster. Pivotal HDB supports PL/Python, PL/Java and PL/R programming languages.

3.4. Flexible Data Format Support

HDB supports multiple data file formats including Apache Parquet and HDB binary data files, plus HBase and Avro via HDB's Pivotal Extension Framework (PXF) services. HDB interfaces with HCatalog, which enables you to query an even broader range of data formats.

3.5. Tight Integration with Hadoop Ecosystem

Pivotal HDB plugs into the Apache Ambari[11] installation, management and configuration framework. This provides a Hadoop-native mechanism for installation and deployment of Pivotal HDB and for monitoring cluster resources across Pivotal HDB and the rest of the Hadoop ecosystem.

4. ECOSYSTEM

HAWQ uses Hadoop ecosystem[12] integration and manageability and flexible data-store format support. HAWQ is natively in hadoop and requires no connectors.Hadoop Common contains libraries and utilities needed by other Hadoop modules. HDFS is a distributed file-system that stores data on commodity machines, providing very high aggregate bandwidth across the cluster. Hadoop YARN is a resource-management platform responsible for managing computing resources in clusters and using them for scheduling of users applications. Hadoop MapReduce is an implementation of the MapReduce programming model for large scale data processing.

5. APPLICATIONS OF PIVOTAL HD/HAWQ

The Pivotal HD Enterprise product enables you to take advantage of big data analytics without the overhead and complexity of a project built from scratch. Pivotal HD Enterprise is Apache Hadoop that allows users to write distributed processing applications for large data sets across a cluster of commodity servers using a simple programming model.

5.1. Content-Based Image Retrieval using Pivotal HD with HAWQ

Manual tagging is infeasible for image databases of this size, and is prone to errors due to users' subjective opinions.Given a query image, a CBIR[13] system can be potentially used to auto-tag (label) similar images in the collection, with the assigned label being the object category or scene description label. This technology also has an important role to play within a number of non-consumer domains. CBIR systems can be used in health care contexts for case-based diagnoses. A common example is image retrieval on large image databases such as Flickr[14].

5.2. Transition of Hulu from Mysql to Pivotal-HD/HAWQ

Hulu is a leading video company that offers TV shows, clips, movies and more on the free, ad-supported Hulu.com service and the subscription service Hulu Plus. It serves 4 billion videos. It has used HAWQ to gain performance improvement to handle queries from users. It's main challenge was inability to scale MySQL and Memcached to improve performance which was handled by Pivotal HAWQ[15].

6. DISADVANTAGES

There are also some drawbacks that needs attention before one choose to use Pivotal-HD/HAWQ. Since most of these are used with the help public cloud providers there is a greater dependency on service providers, Risk of being locked into proprietary or vendor-recommended systems, Potential privacy and security risks of putting valuable data on someone else's system. Another important problem what happens if the supplier suddenly stops services. Even with this disadvantages the technology is still used greatly in various industries and many more are looking forward to move into cloud.

7. EDUCATIONAL MATERIAL

Pivotal offers an portfolio of role-based courses and certifications to build your product expertise[16]. These courses can get someone with basic knowledge of hadoop ecosystem to understand how to deploy, manage, build, integrate and analyze Pivotal HD/HAWQ applications on clouds.

8. CONCULSION

Pivotal HD/HAWQ is the Apache Hadoop native SQL database for data science and machine learning workloads. With Pivotal HDB we can ask more questions on data in Hadoop to gain insights into data by using all the available cloud resources without sampling data or moving it into another platform for advanced analytics. Its fault tolerant architecture can handle node failures and move the workload around clusters.

REFERENCES

[1] "About hawq," webpage. [Online]. Available: http://hdb.docs.pivotal.io/212/hawq/overview/HAWQOverview.html

[2] M. Rouse and M. Haughn, "About mpp," webpage. [Online]. Available: http://searchdatamanagement.techtarget.com/definition/MPP-database-massively-parallel-processing-database

[3] "About udp," webpage. [Online]. Available: https://en.wikipedia.org/wiki/User_Datagram_Protocol

[4] "About nodemanager overview," webpage. [Online]. Available: https://docs.oracle.com/cd/E13222_01/wls/docs81/adminguide/nodemgr.html

[5] "Architecture of pivotal hawq," webpage. [Online]. Available: http://hdb.docs.pivotal.io/212/hawq/overview/HAWQArchitecture.html

[6] "Pivotal architecture online image," webpage. [Online]. Available: https://hdb.docs.pivotal.io/211/hawq/mdimages/hawq_high_level_architecture.png

[7] "About jdbc," webpage. [Online]. Available: https://en.wikipedia.org/wiki/Java_Database_Connectivity

[8] "About odbc," webpage. [Online]. Available: https://en.wikipedia.org/wiki/Open_Database_Connectivity

[9] "About udt," webpage. [Online]. Available: https://en.wikipedia.org/wiki/UDP-based_Data_Transfer_Protocol

[10] "About olap," webpage. [Online]. Available: https://en.wikipedia.org/wiki/Online_analytical_processing

[11] "About apache ambari," webpage. [Online]. Available: https://ambari.apache.org/

[12] "About hadoop," webpage. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Hadoop

[13] "About cbir," webpage. [Online]. Available: https://content.pivotal.io/blog/content-based-image-retrieval-using-pivotal-hd-with-hawq

[14] E. Hörster, R. Lienhart, and M. Slaney, "Image retrieval on large-scale image databases," in *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, ser. CIVR '07. New York, NY, USA: ACM, 2007, pp. 17–24. [Online]. Available: http://doi.acm.org/10.1145/1282280.1282283

[15] "pivotal hulu use case," webpage. [Online]. Available: http://hdb.docs.pivotal.io/212/hawq/overview/HAWQArchitecture.html

[16] "educational courses," webpage. [Online]. Available: https://pivotal.io/training/courses

An overview of Cisco Intelligent Automation for Cloud

BHAVESH REDDY MERUGUREDDY[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: bmerugur@umail.iu.edu*

Paper2, April 30, 2017

Cisco Intelligent automation for cloud is a cloud platform that can deliver services across mixed environments. Its services range from underlying infrastructure to anything-as-a-service and allows the users to evaluate, transform and deploy various IT services. It provides a foundation for organizational transformation by expanding the uses of cloud technology beyond its infrastructure.

Keywords: Automation, Multitenancy, Integration, Orchestration, Provisioner

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2018/report.pdf

1. INTRODUCTION

Cisco Intelligent Automation for Cloud (IAC) provides a framework that allows users to make use of the cloud services effectively and manage the cloud beyond Infrastructure-as-a-service (IaaS) [1]. It can be considered as a unified cloud platform that can deliver any type of service across mixed environments [2]. This leads to an increase in cloud penetration across different businesses. Its services range from underlying infrastructure to anything-as-a-service by allowing its users to evaluate, transform and deploy the IT and business services in a way they desire. Cisco Intelligent Automation for Cloud automates sophisticated data center from a single self-service portal which is beyond the provision of virtual machines. It creates a catalog of standardized service offerings, thereby implementing policy-based controls. It also provides resource management across different aspects of IT infrastructure such as network, virtualization, storage, compute and applications.

2. COMPONENTS

Cisco Intelligent Automation for Cloud consists of major interface and automation components. In other words, it provides an integrated stack of core elements namely Cisco Cloud Portal, Cisco Process Orchestrator, Cisco Process Orchestrator Integration Framework, Cisco Server Provisioner, and Cloud Automation Packs [3].

2.1. Cisco Cloud Portal

Cisco Cloud Portal is a web-based service portal that helps users to order and manage services. It provides a configurable interface for different roles and departments which include managers, administrators and consumers [3]. This provides an access to a catalog of on-demand IT resources. Service requests for resources running on cross-platform infrastructure can be managed effectively by Cisco Cloud Portal. It tracks and manages lifecycle of each service from the initial request to withdrawal. The modules which constitute Cisco Cloud Portal are Cisco Portal Manager, Cisco Service Catalog, Cisco Request Center and Cisco Service Connector.

Cisco Portal Manager combines data from multiple sources providing a highly flexible portal interface. It manages the interface and increases user satisfaction. It provides a drag-and-drop facility on the interface making it easier for the users to create their own portal views which ensures flexibility [4]. Depending on the user requests, Cisco Service Catalog provides a controlled access to IT resources through standardized service options. It makes use of reusable components and tools for publishing services in the portal. Cisco Request Center provides lifecycle management and request management for the infrastructure services. To ensure data center management, it maintains policy-based controls and reduces the cycle time for the ordering, approval and provisioning process. It uses advanced methods for simplifying the ordering process. It also helps in streamlining end-to-end service delivery cycle time. Cisco Service Connector is a platform that can be used for integrating with third-party systems. This supports a heterogeneous data center environment by providing adaptors for integrating with automated provisioning systems.

2.2. Cisco Process Orchestrator

Cisco Process Orchestrator is a component of Cisco IAC responsible for automation of service management and assurance instantiation. It is an orchestration engine that provides an automation design studio and a reporting and analytics module. It is useful in automating and standardizing IT processes in heterogeneous environments [5]. It considers IT automation as a service-oriented approach and focuses more on services. It allows users to deploy services by defining new instances in

real time. Cisco Process Orchestrator acts as a foundation for building application, network and data center oriented solutions. It includes auditing and extensive reporting and provides workspaces for developers and administrators making it easier for the stakeholders to manage services.

The primary function of Cisco Process Orchestrator is to provide automation through integration with domain managers and tools in the environment [6]. Automation Packs and Adapters are the features used in the integration. Some of the services include event correlation, application provisioning and event application provisioning. It usually receives events requiring further analysis. It also includes security tools, configuration tools, change management systems, visualization management tools, provisioning systems and service desks.

2.3. Cisco Process Orchestrator Integration Framework

Cisco Process Orchestrator Integration Framework is responsible for integrating Cisco IAC with any data center element in the environment. It connects IT service management tools into streamlined and automated processes by making use of field-built integration. VMware, SAP, Oracle DB, Remedy and Windows are some of the available integrations. Field integration and automation are carried out by the design studio which provides web services, database access and command-line interface.

2.4. Cisco Server Provisioner

Cisco Server Provisioner is a software application used for deploying systems with Linux, Windows and Hypervisors from bare metal in IT organizations and data centers [7]. It acts as an application for native and remote installations on physical and virtual servers. It provides imaging component for OS and Hypervisor. The Cisco Server Provisioner can be considered as a server suite that consists of Bare Metal Provisioning Payloads and Bare Metal Imaging Payloads which provide GUI and API interfaces including remote file copying and remote troubleshooting. Provisioner runs on a dedicated system which does not run any other application. Some of the benefits of the Provisioner include reduced deployment time and increased utilization of systems.

2.5. Cloud Automation Packs

Cloud Automation Packs are the workflows useful for complex computing tasks. The tasks include automation of core activities that cover various domains, Cisco Server Provisioner task automation, VMware task automation and Cisco UCS Manager task automation. Automation Packs provide a set of target groups, variables, configurations and process definitions required for defining automated IT processes. Automation Packs combine with Adaptors to enable integrations. Integration with IT element is carried out through a combination of automation content from an Automation Pack and a set of Adaptors. Some of the integration scenarios include Command Line Interface invocations, Web Service integration, Messaging integration and Scripting support. Automation content can leverage Adaptors to enable these scenarios. Automation Packs can be used to build, pack, update and ship the integrations.

3. FUNCTIONS

Cisco Intelligent Automation for Cloud provides a platform for designing, deploying and operating a cloud infrastructure in a public, private or a hybrid model. It supports various cloud management activities including setup and design, system operations, reporting and analytics.

3.1. Self-Service Interface

Self-Service Interface is a web-based interface which allows the users to view the service catalog according to their roles and other access controls. It provides dynamic forms by which users can provide configuration details and order services. It also allows the users to track order status, manage and modify placed orders and view the usage and consumption.

3.2. Service Delivery Automation

After the approval of the placed orders, Service Delivery Automation takes place to orchestrate the configuration and provision of resources like compute, network, storage and supporting services such as firewall, disaster recovery and load balancing [3]. The automated provisioning provides consumption tracking and integration into metering and billing systems. The automated processes then orchestrate the configuration updates and allows the service information updates to be sent back to the system management tools and web-based portal.

3.3. Operational Process Automation

Operational Process Automation coordinates the operational tasks for cloud management which include service-level management, alerting, reporting, capacity planning, performance management, user management and maintenance checks. It provides user administration capability to control user roles and identity, placing the users securely isolated from each other. Systems incidents can be managed by the users through alert management feature and all the processes and results can be tracked and reported by the reporting functionality. Operational Process Automation consists of an Automation Control Center which is a console for viewing and controlling automated processes.

3.4. Network Automation

Cisco IAC allows a manual pre-provisioning of network layer with the increasing amount of data being placed in cloud. This is carried out by Network Automation. Network Automation enables deployment of network services through the Self-Service Portal with a single order [8]. It facilitates dynamic installation of virtual network devices with onboarding users and creates network topologies for the users based on the applications they use.

3.5. Cloud Governance

Through Cloud Governance, Cisco IAC delivers a set of measures for tracking business-oriented metrics. Cloud Governance provides portfolio management across different cloud environments. It allows organizations to establish limits ahead of time. Financial granularity is achieved by these consumption limits.

3.6. Advanced Multitenancy

Advanced Multitenancy allows multiple users to securely reside in a shared environment by providing isolated containers. User management work is offloaded from the cloud provider as the users are controlled by the cloud administrators. Multitenancy sets service pricing per user and accordingly, the services can be enabled or disabled per organization or user. It provides onboarding, modification and offboarding of users and enables secure containers by instantiating network devices.

3.7. Multicloud Management

By Multicloud Management, service providers can tailor their services to specific project needs and specific functions of the hypervisor platforms. For example, Cisco IAC supports two infrastructure layers namely Cisco UCS Director and OpenStack [8]. It allows administrators to manage network services and virtual machines by integrating with Havana and Icehouse. This allows the Cisco UCS Director users to manage Microsoft System Center Virtual Machine Manager.

3.8. Resource Management

Cisco IAC Resource Management orchestrates resource-level operations across different hypervisors such as Hyper-V, Xen or VMware, compute resources such as Cisco UCS, network resources and storage resources such as NetApp. This is done by orchestrating the requests to domain resource managers. It provides maintenance and replacement of units. Capacity management is provided by automated capacity utilization checks, trending reports and alerts. The usage and user quota are managed by automated monitoring and metering of user accounts.

3.9. Lifecycle Management

Lifecycle Management is responsible for creation of service definitions which include selection parameters, business and technical processing flows, pricing options and design descriptions. It also involves management of a service model and underlying automation design for managing a service. Automation design is managed by creating workflows to automate service provisioning, modification, decommissioning and upgrades. The designs are modified by point-and-click-tools instead of custom programming. Lifecycle Management tracks all aspects of services that are running which includes business and project information.

4. DEPLOYMENT

Cisco Intelligent Automation for Cloud is deployed as a software solution. For planning, preparation, design, implementation and optimization of cloud services, it is deployed along with services engagement. The services engagement involves creation of automation workflows and development of a cloud strategy. It focuses on service capabilities for the software deployment.

5. CONCLUSION

Cisco Intelligent Automation for Cloud is a framework useful in expanding cloud responsiveness and flexibility. It provides services beyond provisioning virtual machines. It delivers various services in a self-service manner across mixed environments. The cloud management in Cisco Intelligent Automation for Cloud allows users to evaluate, procure and deploy IT services according to the needs. Self-Service Portal, Service Delivery Automation, Network Automation, Resource Management, Provisioning, Advanced Multitenancy, Portfolio Management and Lifecycle Management are the elements responsible for the cloud management in Cisco Intelligent Automation for Cloud. Cisco IAC integrates with Cisco Unified Computing System, one of the Cisco solutions, to provide the required environment for big data and analytics.

6. ACKNOWLEDGEMENTS

The author thanks Professor Gregor Von Lazewski and all the AIs of Big Data class for the guidance and technical support.

REFERENCES

[1] T. Hagay, "Build a better cloud with cisco intelligent automation for cloud," Webpage, 2014. [Online]. Available: https://www.ciscolive.com/online/connect/sessionDetail.ww?SESSION_ID=76497&tclass=popup

[2] S. Miniman, "Cisco moves up the cloud stack with intelligent automation," Webpage. [Online]. Available: http://wikibon.org/wiki/v/Cisco_Moves_Up_the_Cloud_Stack_with_Intelligent_Automation

[3] *Cisco Intelligent Automation for Cloud*, Cisco Systems, 2011. [Online]. Available: http://www.cisco.com/c/dam/en_us/training-events/le21/le34/downloads/689/vmworld/CIAC_Cisco_Intelligent_Automation_Cloud.pdf

[4] *Cisco Cloud Portal*, Cisco Systems, 2011. [Online]. Available: http://www.cisco.com/c/dam/en/us/products/collateral/cloud-systems-management/cloud-portal/cisco_cloud_portal.pdf

[5] "Cisco process orchestrator," Webpage. [Online]. Available: http://www.cisco.com/c/en/us/products/cloud-systems-management/process-orchestrator/index.html

[6] *Cisco Process Orchestrator 3.0 Integrations and Automation Packs*, Cisco Systems, 2013. [Online]. Available: http://www.cisco.com/en/US/docs/net_mgmt/datacenter_mgmt/Process_Orchestrator/3.0/Integration_Automation/CPO_3.0_Integrations_and_Automation_Packs.pdf

[7] *Cisco Server Provisioner User's Guide*, LinMin Corp. and Cisco Systems. [Online]. Available: https://linmin.com/cisco/help/index.html?introduction.html

[8] F. Mondora, "Cisco intelligent automation for cloud 4.0," Webpage. [Online]. Available: https://www.mondora.com/#!/post/12d838f9aa9a6a3c144d29cc6ff56a7c

AUTHOR BIOGRAPHIES

Bhavesh Reddy Merugureddy is pursuing his MSc in Computer Science from Indiana University Bloomington

KeystoneML

VASANTH METHKUPALLI[1,*]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: mvasanthiiit@gmail.com

Paper-2, April 30, 2017

KeystoneML is a software framework, written in Scala, from the UC Berkeley AMPLab designed to simplify the construction of large scale, end-to-end, machine learning pipelines with Apache Spark. KeystoneMl and spark.ml share many features, but however there are a few important differences, particularly around type safety and chaining, which lead to pipelines that are easier to construct and more robust. KeystoneML also presents a richer set of operators than those present in spark.ml including featurizers for images, text, and speech, and provides several example pipelines that reproduce state-of-the-art academic results on public data sets.

Keywords: Cloud, I524, KeystoneML, MLlib, API, Spark

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2019/report.pdf

1. INTRODUCTION

MLlib's goal is to make practical machine learning (ML) scalable and easy. Besides new algorithms and performance improvements that are seen in each release, a great deal of time and effort has been spent on making MLlib easy. Similar to Spark Core, MLlib provides APIs in three languages: Python, Java, and Scala, along with user guide and example code, to ease the learning curve for users coming from different backgrounds. In Apache Spark 1.2, Databricks, jointly with AMPLab, UC Berkeley, continues this effort by introducing a pipeline API to MLlib for easy creation and tuning of practical ML pipelines [1] .

A practical ML pipeline often involves a sequence of data pre-processing, feature extraction, model fitting, and validation stages. For example, classifying text documents might involve text segmentation and cleaning, extracting features, and training a classification model with cross-validation. Though there are many libraries we can use for each stage, working with large-scale datasets is not easy as it looks. Most ML libraries are not designed for distributed computation or they do not provide native support for pipeline creation and tuning. Unfortunately, this problem is often ignored in academia, and it has received largely ad-hoc treatment in industry, where development tends to occur in manual one-off pipeline implementations [1] .

2. DESIGN PRINCIPLES

KeystoneML is built on several design principles: supporting end-to-end workflows, type safety, horizontal scalability, and composibility. By focusing on these principles, KeystoneML allows for the construction of complete, robust, large scale pipelines that are constructed from reusable, understandable parts [2] .

3. KEY API CONCEPTS

At the center of KeystoneML are a handful of core API concepts that allow us to build complex machine learning pipelines out of simple parts:

- Pipelines
- Nodes
- Transformers
- Estimators

4. PIPELINES

A Pipeline is a dataflow that takes some input data and maps it to some output data through a series of nodes. By design, these nodes can operate on one data item (for point lookup) or many data items: for batch model evaluation [2] .

In a sense, a pipeline is just a function that is composed of simpler functions. Here's part of the Pipeline definition:

```
package workflow

trait Pipeline[A, B] {
  // ...
  def apply(in: A): B
  def apply(in: RDD[A]): RDD[B]
  // ...
}
```

Fig. 1. Pipeline Definition

From this we can see that a Pipeline has two type parameters: its input and output types. We can also see that it has methods to operate on just a single input data item, or on a batch RDD of data items [2] .

5. NODES

Nodes come in two flavors:

- Transformers
- Estimators

Transformers are nodes which provide a unary function interface for both single items and RDD of the same type of item, while an Estimator produces a Transformer based on some training data.

5.1. Transformers

As already mentioned, a Transformer is the simplest type of node, and takes an input, and deterministically transforms it into an output. Here's an abridged definition of the Transformer class.

```
package workflow

abstract class Transformer[A, B : ClassTag] extends TransformerNode[B] with Pipeline[A, B] {
  def apply(in: A): B
  def apply(in: RDD[A]): RDD[B] = in.map(apply)
  //...
}
```

Fig. 2. Transformer Class

While transformers are unary functions, they themselves may be parameterized by more than just their input. To handle this case, transformers can take additional state as constructor parameters. Here's a simple transformer which will add a fixed vector from any vector it is fed as input [2] .

```
import pipelines.Transformer
import breeze.linalg._

class Adder(vec: Vector[Double]) extends Transformer[Vector[Double], Vector[Double]] {
  def apply(in: Vector[Double]): Vector[Double] = in + vec
}
```

Fig. 3. Transformer Class-Additional states

5.2. Estimators

Estimators are what puts the ML in KeystoneML. An abridged Estimator interface looks like this:

```
package workflow

abstract class Estimator[A, B] extends EstimatorNode {
  protected def fit(data: RDD[A]): Transformer[A, B]
  // ...
}
```

Fig. 4. Estimator Interface

Estimator takes in training data as an RDD to its fit() method, and outputs a Transformer.Suppose you have a big list of vectors and you want to subtract off the mean of each coordinate across all the vectors (and new ones that come from the same distribution). We could create an Estimator to do this like so.

6. CHAINING NODES AND BUILDING PIPELINES

Pipelines are created by chaining transformers and estimators with the andThen methods. Going back to a different part of the Transformer interface:

Ignoring the implementation, andThen allows you to take a pipeline and add another onto it, yielding a new Pipeline[A,C] which works by first applying the first pipeline (A => B) and then applying the next pipeline (B => C).

This is where type safety comes in to ensure robustness. As your pipelines get more complicated, you may end up trying to chain together nodes that are incompatible, but the compiler won't let you. This is powerful, because it means that if your pipeline compiles, it is more likely to work when you go to run it at scale [3] .

Estimators can be chained onto transformers via the andThen (estimator, data) or andThen (labelEstimator, data, labels) methods. The latter makes sense if you're training a supervised learning model which needs ground truth training labels. Suppose you want to chain together a pipeline which takes a raw image, converts it to grayscale, and then fits a linear model on the pixel space, and returns the most likely class according to the linear model [3] .

7. WHY KEYSTONEML?

KeystoneML makes constructing even complicated machine learning pipelines easy. Here's an example text categorization pipeline which creates bigram features and creates a Naive Bayes model based on the 100,000 most common features [2] .

```
val trainData = NewsGroupsDataLoader(sc, trainingDir)

val predictor = Trim andThen
    LowerCase() andThen
    Tokenizer() andThen
    NGramsFeaturizer(1 to conf.nGrams) andThen
    TermFrequency(x => 1) andThen
    (CommonSparseFeatures(conf.commonFeatures), trainData.data) andThen
    (NaiveBayesEstimator(numClasses), trainData.data, trainData.labels) andThen
    MaxClassifier
```

Fig. 5. Code for Naive Bayes model

Parallelization of the pipeline fitting process is handled automatically and pipeline nodes are designed to scale horizontally. Once the pipeline has been defined you can apply it to test data and evaluate its effectiveness [3] .

```
val test = NewsGroupsDataLoader(sc, testingDir)
val predictions = predictor(test.data)
val eval = MulticlassClassifierEvaluator(predictions, test.labels, numClasses)

println(eval.summary(newsgroupsData.classes))
```

Fig. 6. Code for testing the data for effectiveness

The result of this code is as follows:

```
Avg Accuracy:   0.980
Macro Precision:0.816
Macro Recall:   0.797
Macro F1:       0.797
Total Accuracy: 0.804
Micro Precision:0.804
Micro Recall:   0.804
Micro F1:       0.804
```

Fig. 7. Output for the above code

This relatively simple pipeline predicts the right document category over 80 percent of the time on the test set. KeystoneML works with much more than just text. KeystoneML is alpha software, in a very early public release (v0.2). The project is still very young, but it has reached a point where it is viable for general use.

8. LINKING

KeystoneML is available from Maven Central. It can be used in our applications by adding the following lines to the SBT project definition:

libraryDependencies += "edu.berkeley.cs.amplab"

9. BUILDING

KeystineML is available on GitHub.

$ git clone https://github.com/amplab/keystone.git

Once downloaded, KeystoneML can be built using the following commands:

$ cd keystone
$ git checkout branch-v0.3
$ sbt/sbt assembly
$ make

You can then run example pipelines with the included bin/run-pipeline.sh script, or pass as an argument to spark-submit.

10. RUNNING AN EXAMPLE

Once you've built KeystoneML, you can run many of the example pipelines locally. However, to run the larger examples, you'll want access to a Spark cluster.

Here's an example of running a handwriting recognition pipeline on the popular MNIST dataset. This should be able to run on a single machine in under a minute.

```
#Get the data from S3
wget http://mnist-data.s3.amazonaws.com/train-mnist-dense-with-labels.data
wget http://mnist-data.s3.amazonaws.com/test-mnist-dense-with-labels.data

KEYSTONE_MEM=4g ./bin/run-pipeline.sh \
  pipelines.images.mnist.MnistRandomFFT \
  --trainLocation ./train-mnist-dense-with-labels.data \
  --testLocation ./test-mnist-dense-with-labels.data \
  --numFFTs 4 \
  --blockSize 2048
```

Fig. 8. Example for running on a MNIST dataset

To run on a cluster, it is recommend using the spark-ec2 to launch a cluster and provision with correct versions of BLAS and native C libraries used by KeystoneML.

More scripts have been provided to set up a well-configured cluster automatically in bin/pipelines-ec2.sh.

11. CONCLUSION

One of the main features of KeystoneML is the example pipelines and nodes it provides out of the box. These are designed to illustrate end-to-end real world pipelines in computer vision, speech recognition, and natural language processing. KeystoneML also provides several utilities for evaluating models once they've been trained. Computing metrics like precision, recall, and accuracy on a test set. Metrics are currently calculated for Binary Classification, Multiclass Classification, and Multilabel Classification, with more on the way [3] .

REFERENCES

[1] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen *et al.*, "Mllib: Machine learning in apache spark," *Journal of Machine Learning Research*, vol. 17, no. 34, pp. 1–7, 2016.

[2] "KeystoneML." [Online]. Available: http://keystone-ml.org/programming_guide.html

[3] "KeystoneML." [Online]. Available: http://keystone-ml.org/

Amazon Elastic Beanstalk

SHREE GOVIND MISHRA[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: shremish@indiana.edu

project-000, April 30, 2017

Amazon Elastic Beanstalk is a service provided by the Amazon Web Services which allow developers and engineers to deploy and run web applications in the cloud, in such a way that these applications are highly available and scalable. Elastic Beanstalk manages the deployed application by reducing the management complexities as it automatically handles the capacity provisioning, load balancing, scaling, and application health monitoring. Elastic BeanStalk also provisions one or more AWS Resources such as Amazon EC2 instances when an application is deployed.

Keywords: Cloud, I524, Amazon Web Services, AWS Elastic Beanstalk, Load Balancing, Cloud Watch, AWS Management Console, Auto Scaling

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IR-2021/report.pdf

1. INTRODUCTION

Cloud Computing can be defined as an abstraction of services from infrastructure (i.e. hardware), platforms, and applications (i.e. software) by virtualization of resources [1]. The different form of cloud computing services include IaaS, PaaS, and SaaS which stands for (Infrastructure-as-a-Service), (Platform-as-a -Service), and (Software-as-a-Service) respectively. Elastic Beanstalk is a PaaS offered from the AWS that allows users to create applications and push them over the cloud, wherein creating an environment where the features and services of AWS are used such as Amazon EC2, Amazon RDS, etc. to maintain and scale the application without the need for continuous monitoring.

AWS Elastic Beanstalk is one of the many services provided by the AWS with its functionality to manage the Infrastructure. Elastic beanstalk provides a quick deployment and management of the applications on the Cloud as it automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring. Elastic beanstalk uses highly reliable and scalable services [2].

2. NEED FOR AWS ELASTIC BEANSTALK

Cloud Computing shifts the location of resources to the cloud to reduce the cost associated with over-provisioning, under-provisioning, and under-utilization [3]. When more than required resources are available it is called over-provisioning. When the resources are not used adequately it is known as under-utilization. When the resources available are not enough is called under-provisioning. With Elastic Beanstalk, users can prevent the under-provisioning, under-utilization and over-overprovisioning of computing resources, as it keeps a close check on how the workload for the application is varying with time. The information thus obtained, helps in automatically scaling the application up and down by provisioning the adequate required resources to the application. Elastic Beanstalk also reduces the average response time for the application as it automatically provides the needed resources without manual monitoring and decision making. A user uses one or more of these three scaling techniques to manage and scale the application:

2.1. Manual Scaling in Cloud Environments

In traditional applications, scalability is achieved by predicting the peak loads, then purchasing, setting up and configuring the infrastructure that could handle this peak load [4]. With Manual Scaling, the resources are provisioned at the deployment time and the application servers are added to infrastructure manually, thus there is high latency.

2.2. Semi-automatic Scaling in Cloud Environments

In Semi-automatic Scaling the resources are provisioned dynamically (i.e. at runtime) and automatically (i.e. without user intervention) [1]. Thus, the mean time until when the resources are provisioned are short. However, manual monitoring of resources are still necessary. Since resources are provisioned by request, the problem of the unavailability of an application at peak loads is not completely eliminated.

2.3. Automatic Scaling in Cloud Environments

Elastic Beanstalk uses the Automatic Scaling which overcomes the drawbacks of both the Manual Scaling as well as Semi-automatic Scaling by allowing the users to closely follow the

workload curve of their application, by provisioning of the resources on demand. The user owns the choice that the number for resources these applications are using increases automatically during the time when the demand of resources are high to handle the peak load and also automatically decreases when the minimal resources are needed, to minimize the cost so that the user only pays for what they used. Automatic Scaling also predicts the peak load that the applications may require in the future and provisions these required resources in advance, proving the elasticity of the cloud [1].

3. ELASTIC BEANSTALK SERVICES

Elastic Beanstalk supports applications developed in Java, PHP, NET, Node.js, Python, and Ruby, and also in different container types for each language. A container defines the infrastructure and software stack to be used for a given environment. When an application is deployed, Elastic Beanstalk provisions one or moreAWS resources, such as Amazon EC2 Instances [5]. The software stack that runs on the instances depends on the container type, where two container types are supported by the Elastic Beanstalk Node.js: a 32-bit Amazon Linux Image and a 64-bit Amazon Linux Image. Where each of them runs the Software stack tailored to the hosted Node.js application. Amazon Elastic Beanstalk can be interacted using the AWS Management Console, the AWS Command Line Interface(AWS CLI), or a high-level CLI designed for Elastic BeanStalk [5].

The following web service and features of AWS are used by the Amazon Elastic Beanstalk for the Automatic Scaling of Applications:

3.1. AWS Management Console

AWS Management Console is a browser-based graphical user interface(GUI) for Amazon Web Services. It allows users to configure an automatic scaling mechanism of AWS Elastic Beanstalk as well as other services of AWS. From the Management console, the user can decide about how many instances does the application require. When the application must be scaled up and down [1].

3.2. Elastic Load Balancing

Amazon Elastic Beanstalk uses the Elastic Load Balancing service which enables the load balancer to automatically distributes the incoming application traffic across all running instances in the auto-scaling group based on metrics like request count and latency tracked by Amazon Cloud Watch. If an instance is terminated, the load balancer will not route requests to this instance anymore. Rather, it will distribute the requests across the remaining instances [6].

3.3. Auto Scaling

Auto Scaling automatically launches and terminates instances based on metrics like CPU and RAM utilization of the application and are tracked by Amazon CloudWatch and thresholds called triggers. Whenever a metric crosses a threshold, a trigger is fired to initiate automatic scaling. For example, a new instance will be launched and registered at the load balancer if the average CPU utilization of all running instances exceeds an upper threshold

3.4. Amazon Cloud Watch

Amazon CloudWatch enables the application to monitor, manage and publish various metrics. It also allows configuring alarms based on the data obtained from the metrics to make operational and business decisions. Elastic Beanstalk automatically monitors and scales the application using the CloudWatch.

4. AMAZON ELASTIC BEANSTALK DESCRIPTION

An Elastic Beanstalk Application is a collection of Elastic Beanstalk components which include the application versions, environments, and the environment configurations. Application Version is a deployable code for the web application and it alson implements the deployable code via Amazon Simple Storage Service (Amazon S3) which contains the code. An application may have many different application versions [7].

An environment is a version that is deployed onto AWS resources. At the time of environment creation, Elastic Beanstalk provisions the resources needed to run the application version specified. Where, the environments include an environment tier, platform, and environment type. The environment tier chosen determines whether Elastic Beanstalk provisions resources to support a web application that handles HTTP(S) requests or web application that handles the background processing task. AWS resources created for an environment include one elastic load balancer, an Auto Scaling group, and one or more Amazon EC2 instances [8].

The software stack that runs on the Amazon EC2 instances is dependent on the container type. Where a container type defines the infrastructure topology and software stack to be used for that environment. For example, an Elastic Beanstalk environment with an Apache Tomcat container uses the Amazon Linux operating system, Apache web server, and Apache Tomcat software. In addition, a software component called the host manager. HM runs on each Amazon EC2 server instance. The host manager reports metrics, errors and events, and server instance status, which are available via the AWS Management Console, APIs, and CLIs [8]. The important aspects of Elastic Beanstalk such as security, management version updates, and database and storage are discussed below:

4.1. Security

The application on the Elastic Beanstalk cloud is available publically at myapp.elasticbeanstalk.com for anyone to access. The user can control what other incoming traffic, such as SSH, is delivered or not to your application servers by changing the EC2 security group settings [9]. The IAM (Identity and Access Management) allows the user to manage users and groups in a centralized manner, such as the user can control which IAM users have access to AWS Elastic Beanstalk, and limit permissions to read-only access to Elastic Beanstalk for operators who should not be able to perform actions against Elastic Beanstalk resources [9].

4.2. Management Version Updates

The user can opt to update the AWS Elastic Beanstalk environment automatically to the latest version of the underlying platform running the application during a specified maintenance window [9]. The managed platform updates use an immutable deployment mechanism to perform these updates, where the applications will be available during the maintenance window and consumers will not be impacted by the update.

4.3. Database and Storage

AWS Elastic Beanstalk stores the application files, deployable codes, and server log files in Amazon S3. If the user is using the

AWS Management Console, the AWS Toolkit for Visual Studio, or AWS Toolkit for Eclipse, an Amazon S3 bucket will be created in the user's account for the user, and the files uploaded by the user will be automatically copied from your local client to Amazon S3 [9]. AWS Elastic Beanstalk does not restrict the user to use any particular data persistence technology. The user can choose to use Amazon Relational Database Service (Amazon RDS) or Amazon DynamoDB, or use Microsoft SQL Server, Oracle, or other relational databases running on Amazon EC2. The user can configure AWS Elastic Beanstalk environments to use different databases by specifying the connection information in the environment configuration. The connection string will be extracted from the application code and thus Elastic Beanstalk can configure different databases to work together [9].

5. CONCLUSION

"There is an observation that in many companies the average utilization of application servers ranges from 5 to 20 percent, meaning that many resources like CPU and RAM are idle at peak times" [10]. Thus, Amazon Elastic Beanstalk helps to provide automatic scaling of the application which efficiently utilizes the expensive computing resources and makes the application able to manage the peakload at all times [9].

Other platforms like Google App Engine also let users have their applications to automatically scale both up and down according to demand but with even more restrictions on how users should develop their applications [11]. Wheras, with AWS Elastic Beanstalk there is more control with the user as the user can manage all the elements of the infrastructure or choose to go with Automatic Scaling.

REFERENCES

[1] D. Bellenger, J. Bertram, A. Budina, A. Koschel, B. Pfänder, C. Serowy, I. Astrova, S. G. Grivas, and M. Schaaf, "Scaling in cloud environments," *ACM Digital Library*, pp. 145–150, 2011. [Online]. Available: http://www.wseas.us/e-library/conferences/2011/Corfu/COMPUTERS/COMPUTERS-23.pdf

[2] Amazon Web Services, "Waht is aws elastic beanstalk," Web page, 2017, online; accessed 19-Mar-2017. [Online]. Available: http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/Welcome.html

[3] S. Tai, J. Nimis, C. Baun, and M. Kunze, *Cloud Computing: Web-Based Dynamic IT Services 1st*. Springer Publishing Company, Incorporated ©2011. [Online]. Available: http://www.springer.com/us/book/9783642209161

[4] J. Yang, J. Qiu, and Y. Li, "A profile-based approach to just-in-time scalability for cloud applications." IEEE Computer Society Washington, DC, USA ©2009, 2009, pp. 9–16.

[5] J. Vlie, F. Paganelli, S. van Wel, and D. Dowd, *Elastic Beanstalk*, 1st ed. O'Reilly Media, Inc., 2011, online; accessed 18-Mar-2017.

[6] Amazon Web Services, "Elastic load balancing," Web page, online; accessed 20-Mar-2017. [Online]. Available: https://aws.amazon.com/elasticloadbalancing/

[7] Amazon Web Services, "Elastic beanstalk components," Web page, online; accessed 2-Apr-2017. [Online]. Available: http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts.components.html

[8] Amazon Web Services, "Elastic beanstalk architecture," Web page, online; accessed 30-Mar-2017. [Online]. Available: http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts.concepts.architecture.html

[9] Amazon Web Services, "Frequently asked questions," Web page, online; accessed 2-Apr-2017. [Online]. Available: https://aws.amazon.com/elasticbeanstalk/faqs/

[10] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Communications of the acm," *ACM Digital Library*, vol. 53, pp. 50–58. [Online]. Available: http://cacm.acm.org/magazines/2010/4/81493-a-view-of-cloud-computing/fulltext

[11] Google Inc., "Google cloud platformGoogle App Engine," Web page, online; accessed 18-Mar-2017. [Online]. Available: https://cloud.google.com/appengine/

ASKALON

ABHISHEK NAIK[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: ahnaik@indiana.edu*

project-002, April 30, 2017

ASKALON is a Grid application development and computing environment which aims to provide a Grid to the application developers in an invisible format. This will simplify the development and execution of various workflow applications on the Grid. This will not only allow a transparent Grid access but also will allow the high-level composition of workflow applications. ASKALON basically makes use of five services: Resource Manager, Scheduler, Execution Engine, Performance Analysis and Performance Prediction.

Keywords: Cloud, I524, ASKALON

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2022/report.pdf

1. INTRODUCTION

A Grid is basically a combination of hardware and software that provides seamless, reliable, pervasive as well as high-end computation abilities [1]. Grid based development methodologies are used for application development. These Grid-based development methodologies emphasize providing the application developer with a non-transparent (i.e., opaque) grid. ASKALON provides such invisible (i.e., opaque) grid to the application developers. While using ASKALON, the Grid work flow applications are made using Unified Modeling Language (UML) based services [1]. Similarly even XML can be used here. Besides this, it also enables integration of the workflows that have been created programmatically using languages such as XML. ASKALON typically requires some middleware for this.

2. DESCRIPTION

When using ASKALON, the users need to create Grid workflow applications at a higher level of abstraction using the Abstract Grid Workflow Language (AGWL) which is based on XML. This representation of the workflow is later passed on to the middleware layer for scheduling and exection. Figure 1 shows the typical architecture of ASKALON.

As shown in the figure, the major service components are the Resource Manager which is used for management of the various resources, the Scheduler that is used for scheduling the various workflow applications onto the Grid, the Execution engine which provides reliable and fault tolerant execution, the Performance analyser that analyses the performance and detects bottlenecks and the Performance predictor that estimates the execution times. The following paragraphs provide a brief description about these services:

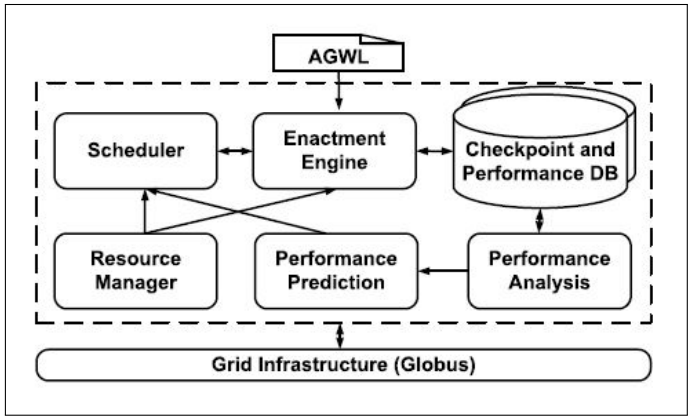


Fig. 1. ASKALON Architecture [1]

2.1. Resource Manager

The Resource Manager service is responsible for the management of the resources like the procurement, allocation, reservation and automatic deployment. It usually works hand in hand with the AGWL. In addition to this management of resources, the main task of the Resource manager is to abstract the users from low-level Grid middleware technology.

2.2. Scheduler

The Scheduler service is responsible for the scheduling (or mapping) of various workflow applications onto the Grid using optimization algorithms and heuristics based on graphs. In addition to this, it monitors the dynamism of the Grid infrastructures thorough an execution contract and adjusts the optimised static

schedules accordingly. The Scheduler thus provides Quality of Service (QoS) dynamically. It usually uses one out of the three algorithms - Heterogeneous Earliest Finish Time (HEFT), Genetic or Myopic. HEFT algorithm schedules the various workflows by creating an ordered lists of tasks and then mapping the tasks onto the resources in the most efficient way. Genetic algorithms are inspired by Darwin's theory of evolution and work using its the princples of evolution while the Myopic algorithms are basically just-in-time approaches, since they make greedy decisions that would be best in the current scenario.

2.3. Execution Engine

The Execution Engine service also known as the Enactment Engine service performs tasks such as checkpointing, retry, restart, migration and replication. It thus aims to provide reliable and fault tolerant execution of the workflows.

2.4. Performance Analyser

The Performance Analyser service provides support to the automatic instrumentation and bottleneck detection within the Grid workflow executions. For example, excessive synchronization, load imbalance or non-scalability of the resources might result in a bottleneck and it is the responsibility of the Performance Analyser to detect and report it.

2.5. Performance Prediction

The Performance Prediction services predicts the performance, i.e., it emphasises on the execution time of the workflow activities. It uses the training phase and statistical methods for this.

3. WORKFLOW GENERATION

ASKALON basically offers two interfaces: graphical model based on UML and a programmatic XML based model [1]. The main aim of both these interfaces is to generate large-scale workflows in a compact as well as intuitive form:

3.1. UML-modeling based

ASKALON allows creation of workflows via a modeling service similar to UML diagrams. This service combines Activity Diagrams and works in a hierarchical fashion. This service can be implemented in a platform independent way using the Model-View Controller (MVC) paradigm. This service can then be shown to contain three parts: a Graphical User Interface (GUI), model traverser and model checker. This GUI in turn comprises of the menu, toolbar, drawing space, model tree and the properties of the elements. The drawing space can contain several diagrams. The model traverser, as the name suggests, provides a way to move throughout the model visiting each element and accessing its properties. The model checker on the other hand is responsible for the correctness of the model.

3.2. XML-based Abstract Grid Workflow Language

The Abstract Grid Workflow Language enables the combination of various atomic units of work called as activities. These activities are interconnected through control-flow and data-flow dependencies. Activities can in turn be of two levels: activity types and activity deployments. An activity type describes the semantics or functions of an activity, whereas the activity deployment points to a deployed Web Service or executable. AGWL is not bounded to any implementation technologies such as the Web Services. Also, the AGWL representation of a typical workflow can be generated in two ways: automatically via the UML representation or manually by the user. In both the cases, AGWL serves as the input to the ASKALON runtime middleware services.

4. COMPARISON WITH OTHER COUNTERPARTS

Few experiments have been carried out using the seven Grid clusters of the Austrian Grid [2] and a group of 116 CPUs. Figure 2 represents performance of the individual clusters, wherein each cluster aggregates the execution time of all the workflows executed on a single CPU. As can be inferred from the figure, the fastest cluster is around thrice faster than the slowest one.

DAGMan [3] is a scheduler that is used for Condor jobs that have been organized in the form of a Directed Acyclic Graph (DAG). Scheduling doesn't use any specialized optimization techniques and is done simply using matchmaking. Fault tolerance is done using rescue DAG that is automatically generated whenever some activity fails. As against this, the ASKALON checkpointing also takes care of the fact when the Execution Engine itself fails. Thus, the checkpointing provided by ASKALON is more robust compared to that by other counterparts like DAGMan.

ASKALON differs in many respects compared to other projects like Gridbus [4] and UNICORE [5]. In ASKALON, the AGWL allows a scalable specification of many parallel activities by using compact parallel loops. The Enactment Engine also enables handling of very large data collections that are generated by large-scale controls and data-flows. The HEFT and genetic search algorithms that the ASKALON scheduler implements, are not used by the other projects, like the ones mentioned above. The Enactment Engine also provides checkpointing of workflows at two leves for restoring and resuming, in case the engine itself fails. In ASKALON, a lot of emphasis is laid on the clear separation between the Scheduler and the Resource Manager. It thus proposes a novel architecture in terms of physical and logical resources and thus provides brokerage, reservations and activity type to deployment mappings.

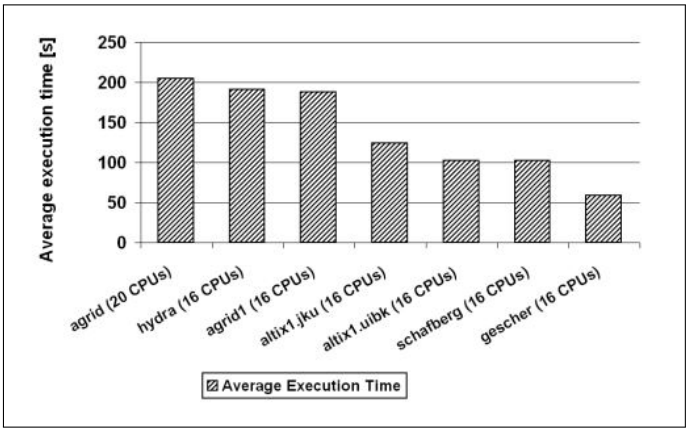


Fig. 2. Performance of Austrian Grid clusters [2]

5. ASKALON IN BIG DATA PROJECTS

ASKALON has been used in the design of Meteorological Stimulations in the Cloud [6]. In order to deploy the application on a distributed Grid infrastructure, the simulation code was split

into a workflow called the RainCloud, which was represented using ASKALON. Figure 3 denotes the graphical representation of this workflow.

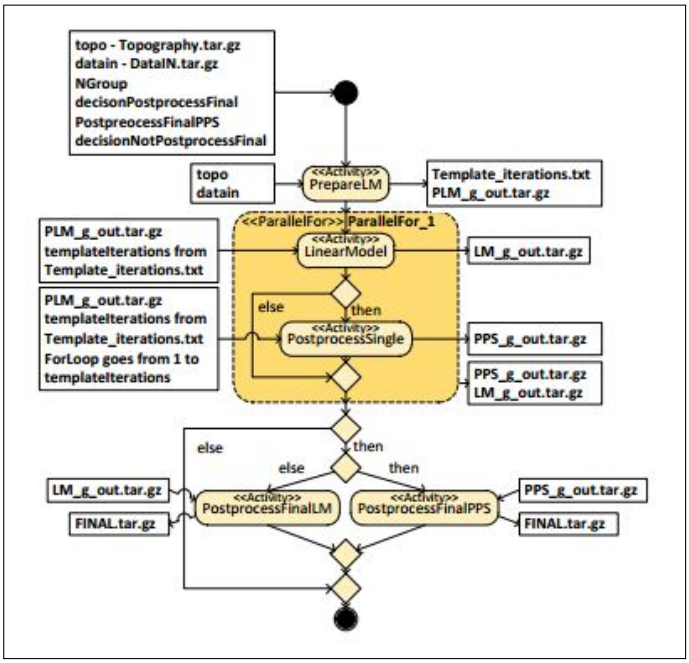


Fig. 3. Graphical representation of the Meteorological workflow [6]

This workflow was flexible in the sense that it could be easy extended to suite the needs of some other projects. For example, this workflow setup was extended for an investigation of precipitation/snow accumulation on the Kongsvegen glacier on Svalbard, Norway, as a part of the SvalClac project [6].

6. CONCLUSION

ASKALON supports workflow integration using UML and also provides an XML based programming interface. This effectively abstracts the end user from the low level middleware technologies. These low level middleware technologies are indeed the ones in charge of scheduling and executing ASKALON applications. The Resource Manager handles the logical and physical resources and the workflow activities to provide features such as authorization, Grid resource discovery, selection, allocation and interaction. Scheduler makes use of some algorithms like HEFT or other genetic algorithms which deliver high performance. It highly benefits from the Performance Prediction service which in turn depend upon the training phase and the statistical methods used. The Execution Engine handles data dependencies and also working on high volume data - something that is highly useful in Big Data related applications and projects. The Performance Analyser analysis the performance benchmarks. Typically, the overhead of ASKALON middleware services which consist of the Resource Manager, Scheduler and Performance prediction are usually constant, thereby requiring less execution time holistically.

Thus, to conclude, we focused on ASKALON as a Grid application development environment. We also saw the various architectural components of ASKALON as well as the comparisons amongst the different Grid clusters that were used. We also saw some Big Data use cases wherein ASKALON was used and the flexibility with which the workflow setup using ASKALON was extended to support different projects.

REFERENCES

[1] I. Taylor, E. Deelman, D. Gannon, and M. Shields, *Workflows for e-Science*. Springer, 2006, pp. 462–471.

[2] "Austrian-grid-project," Web Page, accessed: 2017-3-20. [Online]. Available: http://austriangrid.at

[3] "Dagman-manager," Web Page, accessed: 2017-3-19. [Online]. Available: http://www.cs.wisc.edu/condor/dagman

[4] R. Buyya and S. Venugopal, "Gridbus technologies," in *The Gridbus toolkit for Service Oriented Grid and Utility Computing*, 2004. [Online]. Available: http://www.cloudbus.org/papers/gridbus2004.pdf

[5] "Unicore technologies," Web Page, accessed: 2017-3-20. [Online]. Available: http://www.springer.com/gp/computer-science/lncs

[6] G. Morar, F. Schuller, S. Ostermann, R. Prodan, and G. Mayr, "Meteorological simulations in the cloud with the askalon environment," in *Euro-Par 2012 Parallel Processing Workshops*, vol. 7640, 2012. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-36949-0_9?CFID=916124568&CFTOKEN=29254584

Memcached

RONAK PAREKH[1] AND GREGOR VON LASZEWSKI[2]

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

project-002, April 30, 2017

Memcached is a free and open source, high-performance, distributed memory object caching system. It allows the system to make better use of its memory. It uses data caching technique to speed up dynamic database-driven websites. This reduces the number of times an external data source is accessed by the application. Memcached service is mainly offered through an API. It allows to take memory from parts of the system where you have more memory and allocate it to those parts of the system where you have less memory.

Keywords: Cloud, I524

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2024/report.pdf

1. INTRODUCTION

Memcached is a high-performance, distributed memory object caching system, generic in nature, but originally intended for use in speeding up dynamic web applications by alleviating database load [1]. Memcached allows you to take memory from parts of your system where you have more than you need and make it accessible to areas where you have less than you need. It is a free and open-source software, licensed under the Revised BSD [2] license. Memcached runs on Unix-like operating systems (at least Linux and OS X) and on Microsoft Windows. It depends on the libevent library.

2. COMPARISON OF MEMCACHED AND REDIS

- Server: Memcached server is multi-threaded whereas Redis is single threaded. As a result, Redis [3] can't effectively harness multiple cores.
- Distributability: Memcached is very easy to distribute since it doesn't support any rich data-structures. Redis, on the other hand, is much harder to distribute without changing semantics/performance guarantees of some its commands.
- Ease of Installation: Comparing ease of Installation Redis is much easier. No dependencies required.
- Memory Usage: For simple key-value pairs memcached is more memory efficient than Redis. If you use Redis hashes, then Redis is more memory efficient.
- Persistence: If you are using Memcached then data is lost with a restart and rebuilding cache is a costly process. On the other hand, Redis can handle persistent data. By default, Redis syncs data to the disk at least every 2 seconds.
- Replication: Memcached does not supports replication. Whereas Redis supports master-slave replication. It allows slave Redis servers to be exact copies of master servers.Data from any Redis server can replicate to any number of slaves [4].
- Read/Write Speed: Memcached is very good to handle high traffic websites. It can read lots of information at a time and give you back at a great response time. Redis can also handle high traffic on read but also can handle heavy writes as well.
- Key Length: Memcached key length has a maximum of 250 bytes, whereas Redis key length has a maximum of 2GB. When advanced data structures or disk-backed persistence are important, Redis is used.

3. ARCHITECTURE OF MEMCACHED

Memcached implements a simple and lightweight key-value interface where all key-value tuples are stored in and served from DRAM. The following commands are used by clients to communicate with the Memcached servers:

- SET/ADD/REPLACE(key, value): add a (key, value) object to the cache.
- GET(key): retrieve the value associated with a key.
- DELETE(key): delete a key

Internally, a hash table is used to index the key-value entries. Memcached uses a hash table to index the key-value entries. These entries are also in a linked list sorted by their most recent access time. The least recently used (LRU) entry is evicted and replaced by a newly inserted entry when the cache is full [5].

Hash Table: To lookup keys quickly, the location of each key-value entry is stored in a hash table. Hash collisions are resolved by chaining: if more than one key maps into the same hash table bucket, they form a linked list. Chaining is efficient for inserting or deleting single keys. However, lookup may require scanning the entire chain.

Memory Allocation: Naive memory allocation could result in significant memory fragmentation. To address this problem, Memcached uses slab-based memory allocation. Memory is divided into 1 MB pages, and each page is further sub-divided into fixed-length chunks. Key-value objects are stored in an appropriately sized chunk. The size of a chunk, and thus the number of chunks per page, depends on the particular slab class. For example, by default the chunk size of slab class 1 is 72 bytes and each page of this class has 14563 chunks; while the chunk size of slab class 43 is 1 MB and thus there is only 1 chunk spanning the whole page. To insert a new key, Memcached looks up the slab class whose chunk size best fits this key-value object [5]. If a vacant chunk is available, it is assigned to this item; if the search fails, Memcached will execute cache eviction.

Cache policy In Memcached, each slab class maintains its own objects in an LRU queue (see Figure 1). Each access to an object causes that object to move to the head of the queue. Thus, when Memcached needs to evict an object from the cache, it can find the least recently used object at the tail. The queue is implemented as a doubly-linked list, so each object has two pointers.

Threading: Memcached was originally single-threaded. It uses libevent for asynchronous network I/O callbacks. Later versions support multi-threading but use global locks to protect the core data structures. As a result, operations such as index lookup/update and cache eviction/update are all serialized [5].

4. WORKING OF MEMCACHED

In the Memcached system, each item comprises a key, an expiration time, optional flags, and raw data. When an item is requested, Memcached checks the expiration time to see if the item is still valid before returning it to the client. The cache can be seamlessly integrated with the application by ensuring that the cache is updated at the same time as the database. By default, Memcached acts as a Least Recently Used cache plus expiration timeouts. If the server runs out of memory, it looks for expired items to replace. If additional memory is needed after replacing all the expired items, Memcached replaces items that have not been requested for a certain length of time (the expiration timeout period or longer), keeping more recently requested information in memory.

Working of Memcached Client [6]: Firstly, a memcached client object is created and it starts calling its method to get and set cache entries. When an object is added to the cache, the Memcached client will take that object, serialize it, and send a byte array to the Memcached server for storage. At that point, the cached object might be garbage collected from the JVM where the application is running. When the cached object is needed, the Memcached client's get() method is called. The client will take the get request, serialize it, and send it to the Memcached server. The Memcached server will use the request to look up the object from the cache. Once it has the object, it will return the byte array back to the Memcache client. The client object will then take the byte array and deserialize it to create the object and return it to the application. Even if the application runs on more than one server, all of them can point to the same Memcached server and use it for getting and setting cache entries. The Memcached client is configured in such a way that it knows all the available Memcached servers.

5. SECURITY

Memcached is mostly used for deployments within a trusted network. However, sometimes Memcached is deployed in untrusted networks and enviroments where administrators exercise control over the clients that are connected. SASL authentication support is required for Memcached to compile and it requires the binary protocol. For simplicity, all Memcached operations are treated equally. Clients with a valid need for access to low-security entries within the cache gain access to all entries within the cache. If the cache key can be either predicted, guessed or found by exhaustive searching, its cache entry may be retrieved [7].

6. NEW FEATURES

Chained items were introduced in 1.4.29. With -o modern items sized 512k or higher are created by chaining 512k chunks together. This made increasing the max item size (-I) more efficient in many scenarios as the slab classes no longer have to be stretched to cover the full space. There was still an efficiency hole for items 512k->5mb or so where the overhead is too big for the size of the items. This change fixes it by using chunks from other slab classes in order to "cap" off chained items. With this change larger items should be more efficient than the original slab allocator in all cases.Chunked items are only used with -o modern or explicitly changing -o slab-chunk-max It is not recommended to set slab-chunk-max to be smaller than 256k at this time.

7. USES OF MEMCACHED

Memcached is used for scaling large websites. Facebook is one of the largest users of memcached [8]. It is used to alleviate database load. Facebook uses more than 800 servers supplying over 28 terabytes of memory to our users. As Facebook's popularity had skyrocketed, they've run into a number of scaling issues. This ever increasing demand required Facebook to make modifications to both our operating system and memcached to achieve the performance that provided the best possible experience for their users. There were thousand of computers, each running hundreds of Apache processes which ultimately led to thousands of TCP connections open to the memcached processes. Memcached uses a per-connection buffer to read and write data out over the network. When hundreds and thousands of connections were in consideration, memory added up to be in gigabytes. Memory that could be better used to store user data had to be considered. Thus, memcached came into consideration while scaling large websites.

8. CONCLUSION

Memcached is used when scaling large websites is to be done. By alleviating the database load, it helps making the maximum use of its caching technique to find hits before the database is queried. This results in lesser amount of cycles to and from the external data source or database when it comes to speeding up query results. Memcached has its main advantage in distributability when it comes to setting up a distributed cache and gives superior performance when multithreaded processes are in consideration.

REFERENCES

[1] Dormando, “What is memcached?” Web Page, Mar. 2015, accessed 2017-03-21. [Online]. Available: https://memcached.org/

[2] Wikipedia, “Berkeley software distribution,” Web Page, Mar. 2017, accessed 2017-03-22. [Online]. Available: https://en.wikipedia.org/wiki/Berkeley_Software_Distribution

[3] Wikipedia, “Redis,” Web Page, Mar. 2017, accessed 2017-03-22. [Online]. Available: https://en.wikipedia.org/wiki/Redis

[4] Squarespace Inc., “Memcached vs redis,” Web Page, Mar. 2014, accessed 2017-03-25. [Online]. Available: http://www.openldap.org/lists/openldap-software/200010/msg00097.html

[5] S. M., “Memcached vs redis,” Blog, Mar. 2014, accessed: 23-Mar-2017. [Online]. Available: http://blog.andolasoft.com/2014/02/memcached-vs-redis-which-one-to-pick-for-large-web-app.html

[6] Sunil Patil, “Use memcached for java enterprise performance,” Web Page, Mar. 2012, accessed 2017-03-24. [Online]. Available: http://www.javaworld.com/article/2078565/open-source-tools/open-source-tools-use-memcached-for-java-enterprise-performance-part-1-architecture-and-setup.html

[7] Wikipedia, “Memcached,” Web Page, Feb. 2017, online; accessed 23-Mar-2017. [Online]. Available: https://en.wikipedia.org/wiki/Memcached

[8] Facebook, “Scaling memcached at facebook,” Web Page, Mar. 2015, accessed 2017-03-22. [Online]. Available: https://www.facebook.com/note.php?note_id=39391378919&ref=mf

Naiad

RAHUL RAGHATATE[1,*] AND SNEHAL CHEMBURKAR[1]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: rragahta@iu.edu, snehchem@iu.edu

paper-2, April 30, 2017

Naiad is a distributed system based on computational model called timely dataflow developed for execution of data-parallel, cyclic dataflow programs. It provides an in-memory distributed dataflow framework which exposes control over data partitioning and enables features like the high throughput of batch processors, the low latency of stream processors, and the ability to perform iterative and incremental computations. These features allow the efficient implementation of many dataflow patterns, from bulk and streaming computation to iterative graph processing and machine learning. This paper explains the Naiad Framework, its abstractions, and the reasoning behind it.

Keywords: Naiad, iterative, dataflow, graph, stream

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2026/report.pdf

1. INTRODUCTION

Abundance of distributed dataflow systems around the big data ecosystem has resulted in achieving many goals like high throughput, incremental updating, low latency stream processing, iterative computation. An iterative computation can be think of as some function being looped on its output iteratively until there are no changes between the input and the output and the function converges to a fixed point. On the other hand, in incremental processing, we start with initial input A_0 and produce some output B_0. At some later point, a change δA_1 to the original input A_0, leads to new input $A_1 = A_0 + \delta A_1$. Incremental model produces an incremental update to the output, so $\delta B_1 = F(\delta A_1)$ and $B_1 = \delta B_1 + B_0$. Next state computation in incremental model is based on only previous state (i.e. A0 and B0) [1].

Many data processing tasks require low-latency interactive access to results, iterative sub-computations, and consistent intermediate outputs so that sub-computations can be nested and composed. Most data-parallel systems support either iterative or incremental computations with the dataflow model, but not both. Frameworks like descendants of MapReduce [2, 3],stream processing systems [4, 5], materialized view-maintenance engines [6], provide efficient support for incremental input, but do not support iterative processing. On the other hand, iterative data-parallel frameworks like Datalog [7], recursive SQL databases [8] and systems adding iteration over MapReduce like model [9–12] provide a constrained programming model (e.g. stratified negation in Datalog) and none supports incremental input processing [13].

With a goal to find common low-level abstraction and design general purpose system which resolves above mentioned issues in computation workloads, Naiad, a timely dataflow system was developed at Microsoft by Derek G. Murray *et al.*.

Naiad provides a distributed platform for executing data parallel cyclic dataflow programs. It offers high throughput batch processing, low latency stream processing and the ability to perform iterative and incremental computations all in one framework [14].

In Naiad, the underlying timely dataflow computation framework provides a mechanism for iteration and incremental updates. Moreover, the high-level language support provides a structured means of expressing these computations. This framework can be utilized to build many expressive programming model on top of Naiad's low-level primitives enabling such diverse tasks as streaming data analysis, iterative machine learning, and interactive graph mining [13].

2. ARCHITECTURE

The Naiad architecture consists of two main components- (1) incremental processing of incoming updates and (2) low-latency real-time querying of the application state.

Figure 1 shows a Naiad application that supports real-time queries on continually updated data. The dashed rectangle represents iterative processing that incrementally updates as new data arrive [15].

From Figure 1, query path is clearly separated from the update path. This results in query processing separately on a slightly stale version of the current application state and the query path does not get blocked or incur delays due to the update processing. This also resolves complex situations: If queries shared the same path with updates, the queries could be accessing partially processed/incomplete/inconsistent states, which

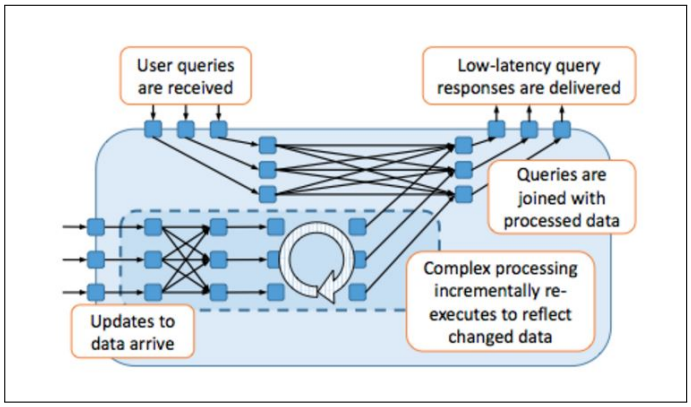


Fig. 1. Naiad Application that support real time queries on continually updated data [15].

would have to be taken care out separately.

Even though hybrid approach to assemble the application in Figure 1 by combining multiple existing systems have been widely deployed, applications built on a single platform as in Figure 1 are typically more efficient, succinct, and maintainable [15].

3. TIMELY DATAFLOW

Applications should produce consistent results, and consistency requires coordination, both across dataflow nodes and around loops [15]. Timely dataflow is a computational model that attaches virtual timestamps to events in structured cyclic dataflow graphs providing coordination mechanism that allows low-latency asynchronous message processing while efficiently tracking global progress and synchronizing only where necessary to enforce consistency. Naiad model simply checkpoints its state periodically, restoring the entire system state to the most recent checkpoint on failure. Even if it is not a sophisticated design, it was chosen in part for its low overhead. Faster common-case processing allows more computation to take place in the intervals between checkpointing, and thus, often decreases the total time to job completion [16].

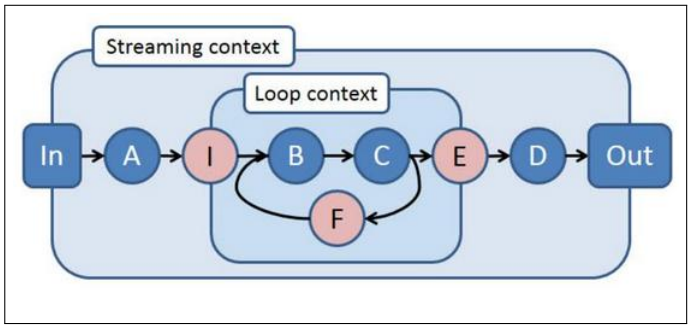


Fig. 2. A simple timely dataflow graph [15].

Consider a simple timely dataflow graph shown in Figure 2 showing a loop context nested in the top-level streaming context. Vertices A and D are not in any loop context resulting in no loop counters for their timestamps. Whereas, vertices B, C, and F are nested in a single loop context, so their timestamps have a single loop counter. The Ingress (I) and Egress (E) vertices sit on the boundary of a loop context. They monitor the loop by adding and removing loop counters to and from timestamps as messages pass through them.

Naiad being a timely dataflow computational model, utilizes similar to above mentioned computational process and provide the basis for an efficient, lightweight coordination mechanism. Timely dataflow supports the following three features:

1. Structured loops allowing feedback in the dataflow
2. Stateful dataflow vertices capable of consuming and producing records without global coordination, and
3. Notifications for vertices once they have received all records for a given round of input or loop iteration.

Structured loops and Stateful dataflow vertices allows iterative and incremental computations with low latency. Notifications makes Naiad possible to produce consistent results, at both outputs and intermediate stages of computations, in the presence of streaming or iteration [15]. The timely dataflow in Naiad is achieved by optimization in services like asynchronous messaging, iterative dataflow, progress tracking and consistency improvisation.

3.1. Asynchronous messaging

All dataflow models require some communication means for message passing between node over outgoing edges. In a timely dataflow system, each node implements an *OnRecv* event handler that the system can call when a message arrives on an incoming edge, and the system provides a *Sendmethod* that a node can invoke from any of its event handlers to send a message on an outgoing edge [16]. Messages are delivered asynchronously.

3.2. Consistency

Computations like reduction functions *Count* or *Average* include subroutines that must accumulate all of their input before generating an output. At the same time, distributed applications commonly split input into small asynchronous messages to reduce latency and buffering. For timely dataflow to support incremental computations on unbounded streams of input as well as iteration, it has a mechanism to signal when a node (or data-parallel set of nodes) has seen a consistent subset of the input for which to produce a result [16].

3.3. Iterative Graph Dataflow

A Naiad dataflow graph is acyclic apart from structurally nested cycles that correspond to loops in the program. The logical timestamp associated with each event represents the batch of input that the event is associated with, and each subsequent integer gives the iteration count of any (nested) loops that contain the node. Every path around a cycle includes a special node that increments the innermost coordinate of the timestamp. The system enforced rule restricts event handler from sending a message with a time earlier than the timestamp for the event it is handling ensuring a *partial order* on all of the pending events (undelivered messages and notifications) in the system, thus enabling efficient progress tracking [15].

3.4. Progress Tracking

The ability to deliver notifications (an event that fires when all messages at or before a particular logical timestamp have been delivered to a particular node) promptly and safely is critical. This provides timely dataflow system an ability to support low-latency incremental and iterative computation with consistent

results. Naiad can implement progress trackers to establish the guarantee that no more messages with a particular timestamp can be sent to a node [15].

4. ACHIEVING TIMELY DATAFLOW

Each event has a timestamp and a location (either a vertex or edge), and are referred as pointstamp. The timely dataflow graph structure ensures that for any locations l_1 and l_2 connected by two paths with different summaries, one of the path summaries always yields adjusted timestamps earlier than the other. For each pair l_1 and l_2, we find the minimal path summary over all paths from l_1 to l_2 using a straightforward graph propagation algorithm, and record it as $\Psi[l_1, l_2]$. To efficiently evaluate the possible relation for two pointstamps (t_1, l_1) and (t_2, l_2), we test whether $\Psi[l_1, l_2](t_1) \leq t_2$ [15].

Informally, Timely Dataflow supports directed dataflow graphs with structured cycles which is analogous to structured loops in a standard imperative programming language. This structure provides information about where records might possibly flow in the computation, allowing an implementation like Naiad to efficiently track and inform dataflow vertexes about the possibility of additional records arriving at given streaming epochs or iterations [17].

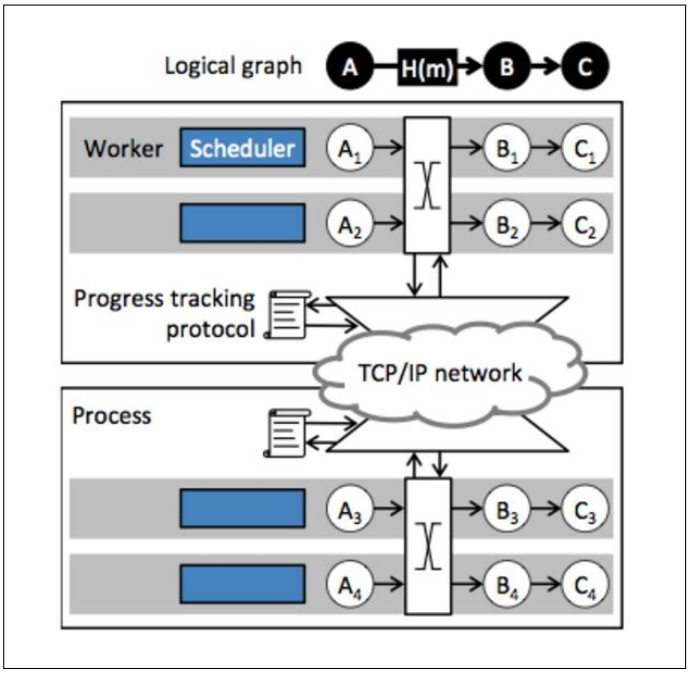


Fig. 3. The mapping of a logical dataflow graph onto the distributed Naiad system architecture [15].

5. SYSTEM IMPLEMENTATION

Naiad is high-performance distributed implementation of timely dataflow and is written in C#, and runs on Windows, Linux, and Mac OS. Figure 3 shows the schematic architecture of a Naiad cluster consisting a group of processes hosting workers that manage a partition of the timely dataflow vertices [16]. Each worker may host several stages of the dataflow graph. The workers are data nodes and keep a portion of the input data (usually a large-scale input graph, such as Twitter follower graph) in memory. So it makes sense to move computation (dataflow graph stages) to the data (the partitioned input graph)[18]. Each process participates in a distributed progress tracking protocol, in order to coordinate the delivery of notifications. All the features of C#, including classes, structs, and lambda functions, to build a timely dataflow graph from a system-provided library of generic Stream objects can be implemented. Naiad model uses deferred execution method: like adding a node to internal data flow graph at runtime while executing a method like Max on a Stream [15].

Naiad's distributed implementation also exhibits features like distributed progress tracking, a simple but extensible implementation of fault tolerance, high availability, avoidance of micro-straggler (events like packet loss, contention on concurrent data structures, and garbage collection which leads to delays ranging from tens of milliseconds to tens of seconds) which is main obstacle to scalability for low-latency workloads [15].

6. PERFORMANCE EVALUATION

Naiad's performance over supporting high-throughput, low latency, data-parallelism, batch processing, iterative graph data processing have been examined using several notable microbenchmarks by Murray *et al.* [15] such as Throughput, Latency, Protocol Optimizations, Scaling.

The hardware configuration consists of two racks of 32 computers, each with two quad-core 2.1 GHz AMD Opteron processors, 16 GB of memory, and an Nvidia NForce Gigabit Ethernet NIC. Also rack switches have 40 Gbps uplink to the core switch. Average across five trials are plotted, with error bars showing minimum and maximum values [15].

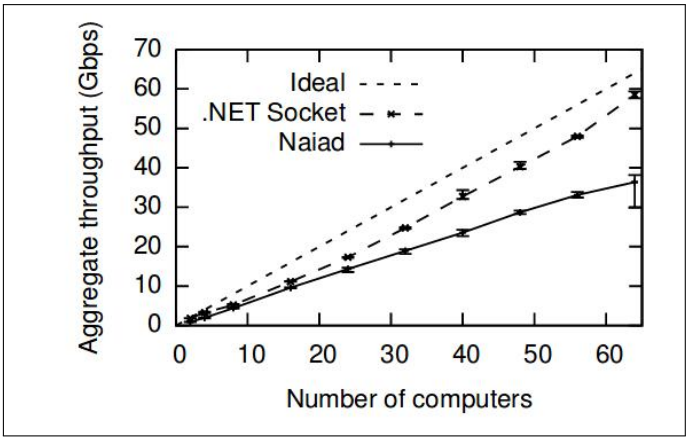


Fig. 4. Evaluation of Naiad system throughput rate on synthetic dataset [15].

6.1. Throughput

The Naiad program constructs a cyclic dataflow that repeatedly performs the all-to-all data exchange of a fixed number of records. Figure 4 plots the aggregate throughput against the number of computers with uppermost line as Ideal throughput, middle one depicting achievable throughput given network topology, TCP overheads, and .NET API costs. The final line shows the throughput that Naiad achieves for a large number of 8-byte records (50M per computer) exchanges between all processes in the cluster.

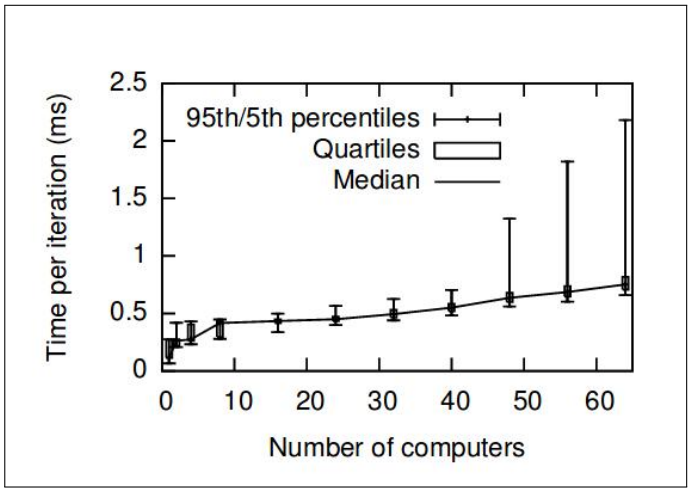


Fig. 5. Evaluation of Naiad system global barrier latency for synthetic dataset [15].

6.2. Latency

Latency microbenchmark evaluates the minimal time required for global coordination. Figure 5 plots the distribution of times for 100K iterations using median, quartiles, and 95^{th} percentile values indicating low median time per iteration and adverse impact at 95^{th} percentile mark due to increased number of computers.

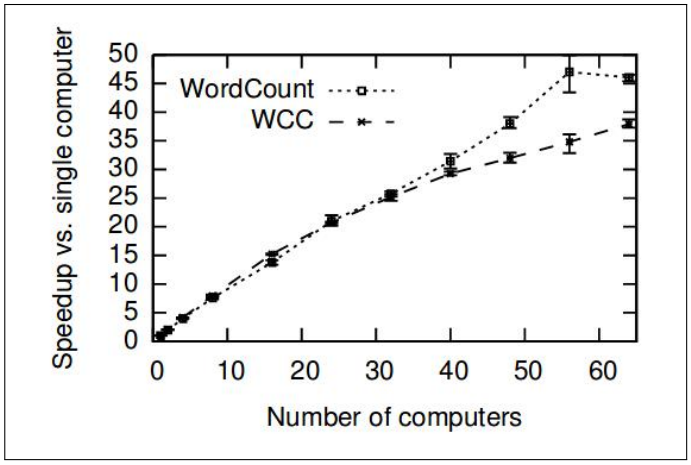


Fig. 6. Strong Scaling microbenchmark evaluation for Naiad system over synthetic dataset [15].

6.3. Scaling

Based on input size and resource availability, strong and weak scaling microbenchmarks are achieved. Keeping input fixed and increased compute resources, running time as shown in Figure 6 has been plotted for two applications, WordCount and Weakly connected components(WCC) . For WordCount, dataset used is Twitter corpus of size 128 GB uncompressed and for WCC, a random graph of 200M edges. Weak Scaling evaluation measured the effect of increasing both the number of computers and the size of the input.

Figure 7 shows the performance of WCC on random input graph with constant number of edges (18.2M) and nodes (9.1M) per computer. The running time degrades significantly to 29.4s

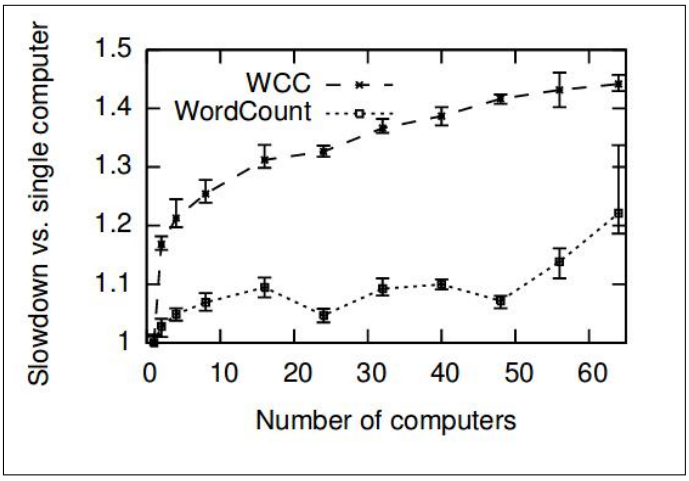


Fig. 7. Weak Scaling microbenchmark evaluation for Naiad system over synthetic dataset [15].

compared to 20.4s for 1.1B edge graph run on 64 computers cluster, relative to execution in single computer. Also the WordCount application, with 2 GB compressed input per computer, doesn't achieved perfect weak scaling, but compared to WCC, it improved.

7. USE CASES

Naiad framework has been majorly deployed for batch, streaming and graph computations serving interactive queries against the results where Naiad responds to updates and queries with sub-second latencies [15].

7.1. Batch iterative graph computation

Implementation of graph algorithms such as PageRank, strongly connected components (SCC), weakly connected components(WCC), and approximate shortest path(ASP) in Naiad requires less code complexity and provides dominant difference in running times compared to PDW [19], DryadLINQ [20],SHS [21] for Category A web graph [15, 22, 23].

7.2. Batch iterative machine learning

Naiad provides competitive platform for custom implementation of distributed machine learning. Also it is straightforward to build communication libraries for existing applications using Naiad's API. For E.g., modified version of Vowpal Wabbit (VW), an open-source distributed machine learning library which performs iterative linear regression [24]. AllReduce (processes jointly performing global averaging) implementation requires 300 lines of code, around half as many as VW's AllReduce, and the Naiad code is at a much higher level, abstracting the network sockets and threads being used [15].

7.3. Streaming acyclic computation

Latency reduction while computing the k-exposure metric for identifying controversial topics on Twitter using Kineograph, which takes snapshots of continuously ingesting graph data for data parallel computations [25].

7.4. Streaming iterative graph analytics

Twitter Analysis: To compute the most popular hashtag in each connected component of the graph of users mentioning other

users, and provide interactive access to the top hashtag in a user's connected component.

8. USEFUL RESOURCES

Naiad: A Timely Dataflow System [15], provides extensive study of Naiad's architecture, methodology of working, its distributed implementation, iterative and incremental data processing, data analysis applications, comparison with other graph processing frameworks. Moreover, [26], [27], [13] are also good resources to learn about Naiad and programming in Naiad Framework.

9. CONCLUSION

Naiad enrich dataflow computations with timestamps that represent logical points in the computational process and provide the basis for an efficient, lightweight coordination mechanism. All the above capabilities in one package allows development of High-level programming models on Naiad which can perform tasks as streaming data analysis, iterative machine learning, and interactive graph mining. Moreover, public reusable low-level programming abstractions of Naiad allow it to outperform many other data parallel systems that enforce a single high-level programming model.

ACKNOWLEDGEMENTS

This work was done as part of the course "I524: Big Data and Open Source Software Projects" at Indiana University during Spring 2017. Many thanks to Professor Gregor von Laszewski and Prof. Geoffrey Fox at Indiana University Bloomington for their academic as well as professional guidance. We would also like to thank Associate Instructors for their help and support during the course.

REFERENCES

[1] Aleksey Charapko, "One page summary: Incremental, iterative processing with timely dataflow," Blog, Feb. 2017, accessed: 09-Apr-2017. [Online]. Available: http://charap.co/one-page-summary-incremental-iterative-processing-with-timely-dataflow/

[2] P. Bhatotia, A. Wieder, R. Rodrigues, U. A. Acar, and R. Pasquin, "Incoop: Mapreduce for incremental computations," in *Proceedings of the 2Nd ACM Symposium on Cloud Computing*, ser. SOCC '11. New York, NY, USA: ACM, 2011, pp. 7:1–7:14, accessed: 2017-3-22. [Online]. Available: http://doi.acm.org/10.1145/2038916.2038923

[3] P. K. Gunda, L. Ravindranath, C. A. Thekkath, Y. Yu, and L. Zhuang, "Nectar: Automatic management of data and computation in datacenters," in *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 75–88, accessed: 2017-3-22. [Online]. Available: http://dl.acm.org/citation.cfm?id=1924943.1924949

[4] R. S. Barga, J. Goldstein, M. H. Ali, and M. Hong, "Consistent streaming through time: A vision for event stream processing," *CoRR*, vol. abs/cs/0612115, 2006, accessed: 2017-3-22. [Online]. Available: http://arxiv.org/abs/cs/0612115

[5] B. Gedik, H. Andrade, K.-L. Wu, P. S. Yu, and M. Doo, "Spade: The system s declarative stream processing engine," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '08. New York, NY, USA: ACM, 2008, pp. 1123–1134, accessed: 2017-3-22. [Online]. Available: http://doi.acm.org/10.1145/1376616.1376729

[6] A. Gupta and I. S. Mumick, "Materialized views," A. Gupta and I. S. Mumick, Eds. Cambridge, MA, USA: MIT Press, 1999, ch. Maintenance of Materialized Views: Problems, Techniques, and Applications, pp. 145–157, accessed: 2017-3-24. [Online]. Available: http://dl.acm.org/citation.cfm?id=310709.310737

[7] S. Ceri, G. Gottlob, and L. Tanca, "What you always wanted to know about datalog (and never dared to ask)," *IEEE Transactions on Knowledge and Data Engineering*, vol. 1, no. 1, pp. 146–166, Mar 1989, accessed: 2017-3-24.

[8] A. Eisenberg and J. Melton, "Sql: 1999, formerly known as sql3," *SIGMOD Rec.*, vol. 28, no. 1, pp. 131–138, Mar. 1999, accessed: 2017-3-22. [Online]. Available: http://doi.acm.org/10.1145/309844.310075

[9] Y. Bu, B. Howe, M. Balazinska, and M. D. Ernst, "Haloop: Efficient iterative data processing on large clusters," *Proc. VLDB Endow.*, vol. 3, no. 1-2, pp. 285–296, Sep. 2010, accessed: 2017-3-22. [Online]. Available: http://dx.doi.org/10.14778/1920841.1920881

[10] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, and G. Fox, "Twister: A runtime for iterative mapreduce," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, ser. HPDC '10. New York, NY, USA: ACM, 2010, pp. 810–818, accessed: 2017-3-23. [Online]. Available: http://doi.acm.org/10.1145/1851476.1851593

[11] D. G. Murray, M. Schwarzkopf, C. Smowton, S. Smith, A. Madhavapeddy, and S. Hand, "Ciel: A universal execution engine for distributed data-flow computing," in *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 113–126, accessed: 2017-3-24. [Online]. Available: http://dl.acm.org/citation.cfm?id=1972457.1972470

[12] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 2–2, accessed: 2017-3-24. [Online]. Available: http://dl.acm.org/citation.cfm?id=2228298.2228301

[13] F. McSherry, R. Isaacs, M. Isard, and D. Murray, "Composable incremental and iterative data-parallel computation with naiad," Tech. Rep. MSR-TR-2012-105, October 2012, accessed: 2017-3-24. [Online]. Available: https://www.microsoft.com/en-us/research/publication/composable-incremental-and-iterative-data-parallel-computation-with-naiad/

[14] HU2LA, "Review of "naiad: A timely dataflow system" < huula . webdesign + ai," Blog, Sep. 2015, accessed: 23-Mar-2017. [Online]. Available: https://huu.la/blog/review-of-naiad-a-timely-dataflow-system

[15] D. G. Murray, F. McSherry, R. Isaacs, M. Isard, P. Barham, and M. Abadi, "Naiad: A timely dataflow system," in *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, ser. SOSP '13. New York, NY, USA: ACM, 2013, pp. 439–455, accessed: 2017-3-22. [Online]. Available: http://doi.acm.org/10.1145/2517349.2522738

[16] D. G. Murray, F. McSherry, M. Isard, R. Isaacs, P. Barham, and M. Abadi, "Incremental, iterative data processing with timely dataflow," *Commun. ACM*, vol. 59, no. 10, pp. 75–83, Sep. 2016, accessed: 2017-3-22. [Online]. Available: http://doi.acm.org/10.1145/2983551

[17] Microsoft, "Naiad-microsoft," Web Page, Microsoft, 2017, accessed: 2017-3-22. [Online]. Available: https://www.microsoft.com/en-us/research/project/naiad/

[18] Edgar, "Metadata and the naiad post – one other good blog on data science/engineering," Blog, Jan. 2017, accessed: 23-Mar-2017. [Online]. Available: https://theinformationageblog.wordpress.com/2017/01/09/metadata-and-the-naiad-post-one-other-good-blog-on-data-scienceengineering/

[19] Microsoft, "Microsoft analytics platform system overview |microsoft," Web Page, Microsoft, 2017, accessed: 2017-3-26. [Online]. Available: https://www.microsoft.com/en-us/sql-server/analytics-platform-system

[20] Y. Yu, M. Isard, D. Fetterly, M. Budiu, U. Erlingsson, P. K. Gunda, and J. Currey, "Dryadlinq: A system for general-purpose distributed data-parallel computing using a high-level language," in *Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 1–14, accessed: 2017-3-24. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855741.1855742

[21] M. Najork, "The scalable hyperlink store," in *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia*, ser. HT '09. New York, NY, USA: ACM, 2009, pp. 89–98, accessed: 2017-3-23. [Online].

Available: http://doi.acm.org/10.1145/1557914.1557933
[22] Lemur, "The clueweb09 dataset," Web Page, Red Hat, Inc., 2017, accessed: 2017-3-24. [Online]. Available: http://lemurproject.org/clueweb09/
[23] M. Najork, D. Fetterly, A. Halverson, K. Kenthapadi, and S. Gollapudi, "Of hammers and nails: An empirical comparison of three paradigms for processing large graphs," in *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, ser. WSDM '12. New York, NY, USA: ACM, 2012, pp. 103–112, accessed: 2017-3-24. [Online]. Available: http://doi.acm.org/10.1145/2124295.2124310
[24] John Lanford, "Home . johnlangford/vowpal_wabbit wiki," Code Repository, Dec. 2016, accessed: 2017-3-24. [Online]. Available: https://github.com/JohnLangford/vowpal_wabbit/wiki
[25] R. Cheng, J. Hong, A. Kyrola, Y. Miao, X. Weng, M. Wu, F. Yang, L. Zhou, F. Zhao, and E. Chen, "Kineograph: Taking the pulse of a fast-changing and connected world," in *Proceedings of the 7th ACM European Conference on Computer Systems*, ser. EuroSys '12. New York, NY, USA: ACM, 2012, pp. 85–98, accessed: 2017-3-22. [Online]. Available: http://doi.acm.org/10.1145/2168836.2168846
[26] Microsoft Research, "Microsoft/naiad:the naiad system provides fast incremental and iterative computation for data-parallel workloads," Code Repository, Nov. 2014, accessed: 2017-3-24. [Online]. Available: https://github.com/MicrosoftResearch/Naiad
[27] naiadquestions@microsoft.com, "Nuget gallery|naiad - core 0.5.0-beta," Web Page, NET Foundation, Nov. 2014, accessed: 2017-3-26. [Online]. Available: https://www.nuget.org/packages/Microsoft.Research.Naiad/

Dryad : Distributed Execution Engine

SHAHIDHYA RAMACHANDRAN[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: shahrama@iu.edu*

Technology paper 2, April 30, 2017

Dryad is a general purpose execution environment for distributed, data parallel applications and it automatically handles job creation and management, resource management, job monitoring and visualization, fault tolerance, re-execution, scheduling, and accounting. It creates a dataflow graph by using computational 'vertices' and communication 'channels'. Dryad is the middleware abstraction that is independent of the semantics of the program. It is not suitable for real-time processing since it focuses on throughput rather than latency.

Keywords: Dryad, Distribute Processing, Concurrent Processing, Dryad LINQ, Dataflow, Cluster Computing

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IR-2027

1. INTRODUCTION

With the expansion of the large scale Internet services that depend on clusters of thousands of servers it became more complicated to manage wide-area distributed computing. Some of the prominent challenges included high-latency, unreliable networks, control of resources by separate federated or competing entities, and issues of identity for authentication and access control [1]. Dryad was launched in an effort to make it easier for developers to write efficient parallel and distributed applications without having in-depth knowledge of concurrent programming [2] It focused primarily on reliability, efficiency and scalability of the applications. Dryad was launched as a research project at MSR(Microsoft Research) and was released in EuroSys'07, March 21–23, 2007, Lisboa, Portugal [3].
A Dryad application is a graph generator which can synthesize any given directed acyclic graph. These graphs are dynamic and can change depending on computations made during runtime [2]. The vertices of the Directed Acyclic Graph define the operations that are to be performed on the data. These 'computational vertices' are written as single threaded sequential C++ constructs and are connected using one-way channels. Channels facilitate communication between the vertices through temporary files, TCP/IP streams, and shared-memory FIFOs [1]. The graph is parallelized by distributing the vertices across multiple execution engines. Dryad can scale this across multiple processor cores on the same computer or different physical computers connected by a network, as in a cluster [1]. Scheduling of the computational vertices on the available hardware is handled by the dryad run-time, without any explicit intervention by the developer of the application or administrator of the network.

2. DRYAD SYSTEM

A Dryad job is a directed acyclic graph where each vertex is a program and edges represent data channels. The logical computation graph is automatically mapped onto physical resources by run-time. The number of vertices in the graph can be much greater than the number of execution cores in the computing cluster. During run-time a finite sequence of 'structured items' are transported through the channels. channels produce and consume heap objects that inherit from a base type. The vertex program then reads and writes its data in the same way irrespective of how the channel serializes its data. Each application has its own serialization/deserialization routine [4].

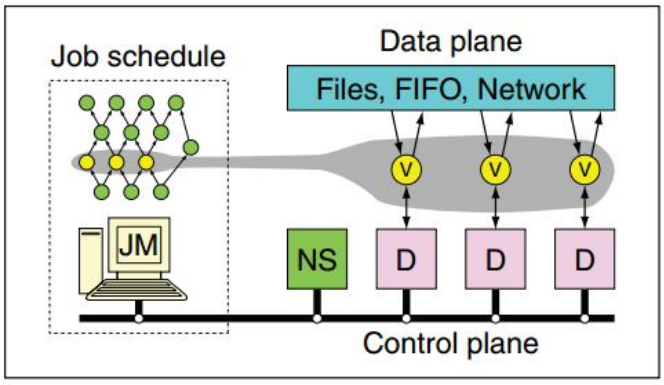


Fig. 1. The Dryad system organization Source:[4]

The overall structure of a Dryad system is shown in the Figure 1. The job manager (JM) consults the name server (NS) to discover

the list of available computers. It maintains the job graph and schedules running vertices (V) as computers become available using the daemon (D) as a proxy. Vertices exchange data through files, TCP pipes, or shared-memory channels. The shaded bar indicates the vertices in the job that are currently running [4].

2.1. Job Manager(JM)

A Dryad job is coordinated by the job manager. It runs either within the cluster or on a user's workstation with network access to the cluster. It contains the code to construct the communication graph and also the library code to schedule the work across the available resources. Job manager handles only the control decisions and is not involved in the actual data transfer between vertices [4].

2.2. Name Server(NS)

The Name Server lists all the available computers and exposes the position of each computer within the network topology so that scheduling decisions can be taken by taking locality into consideration [4].

2.3. Daemon(D)

. It is a cluster that runs on each of the computers and is responsible for creating processes on behalf of the job manager. The first time a vertex (V) is executed its binary is sent from the job manager to the daemon. It acts as a proxy to the job manager when it checks the status of the computation and how much data has been read and written on the channels of the remote vertices [4].

2.4. Job Scheduler

The task scheduler is used to queue batch jobs. A distributed storage system is used that allows large files to be broken into small pieces, replicated and distributed across the local disks of the cluster computers [4].

3. DRYAD GRAPH

Each graph is represented as $G =< V_G, E_G, I_G, O_G >$ [4] where V_G is a sequence of vertices with E_G directed edges and two sets $I_G \subset V_G$ and $O_G \subset V_G$ indicate the input and output vertices respectively. Directed Acyclic graphs are used because a vertex can run anywhere once its inputs have been received, it does not lead to a dead-lock situation and the finite length channels ensure that the process will definitely terminate [5]. Figure 2 shows an example of a Directed Acyclic Graph. One of the primary restrictions on a Dryad job is that it has to be representable as a Directed Acyclic Graph. The graph can have multiple edges between the vertices. Each circle represents a program that processes a set of inputs and outputs the results. The inputs and outputs in the graph are virtual vertices of the distributed Dryad system.

3.1. Inputs and Outputs

Large input files are partitioned and distributed across the computers of the cluster. The input is grouped into a graph $G = V_P, \phi, \phi, V_P$ where V_P [6] is a sequence of virtual vertices corresponding to the partitions of the input. After the job is completed, a set of output partitions are concatenated to form a single named distributed file. An application will generally interrogate its input graphs to read the number of partitions at run-time and automatically generate the appropriately replicated graph.

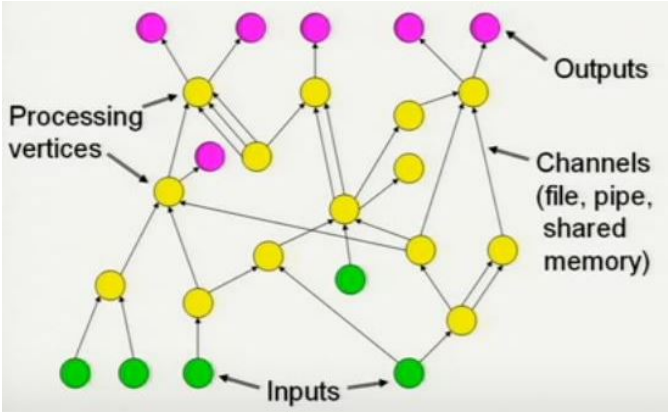


Fig. 2. Directed Acyclic Graph in Dryad job Source:[5]

3.2. Creating new vertices

All vertex programs inherit from the Dryad defined C++ base class. Each program has a textual name - unique within the application and a static factory- that has knowledge of constructing it [6]. A graph vertex is created by calling the appropriate static program factory. Vertex specific parameters are set by calling methods on the program object. These parameters along with the unique vertex name form a closure that is sent to a remote process for execution.

3.3. Adding graph edges

New edges are created by applying a composition operation to two existing graphs as shown in Figure 3. In this figure, Circles are vertices and arrows are graph edges. A triangle at the bottom of a vertex indicates an input and one at the top indicates an output. Boxes (a) and (b) demonstrate cloning individual vertices using the carat operator. The two standard connection operations are pointwise composition using >= shown in (c) and complete bipartite composition using » shown in (d). (e) illustrates a merge using ||. The second line of the figure shows more complex patterns. The merge in (g) makes use of a "subroutine" from (f) and demonstrates a bypass operation. For example, each A vertex might output a summary of its input to C which aggregates them and forwards the global statistics to every B. Together the B vertices can then distribute the original dataset (received from A) into balanced partitions. An asymmetric fork/join is shown in (h) [6].

4. FEATURES

The primary features of Dryad are fault tolerance and dynamic modification of graph during run-time.

4.1. Fault Tolerance

When a vertex execution fails, the job manager will be notified. If the vertex reported an error cleanly/crashes then it is forwarded by the daemon to the job manager. If the daemon also fails then job manager receives a heartbeat timeout. The failed vertex A is re-executed. If vertex A's inputs fail, all upstream vertices are re-executed. vertex A is running slower than it's peers then creates duplicate executions and first output is used [5].

4.2. Dynamic Graph Refinement

The application passes the initial graph at the beginning and records all the callbacks. The graphs can be modified during run-

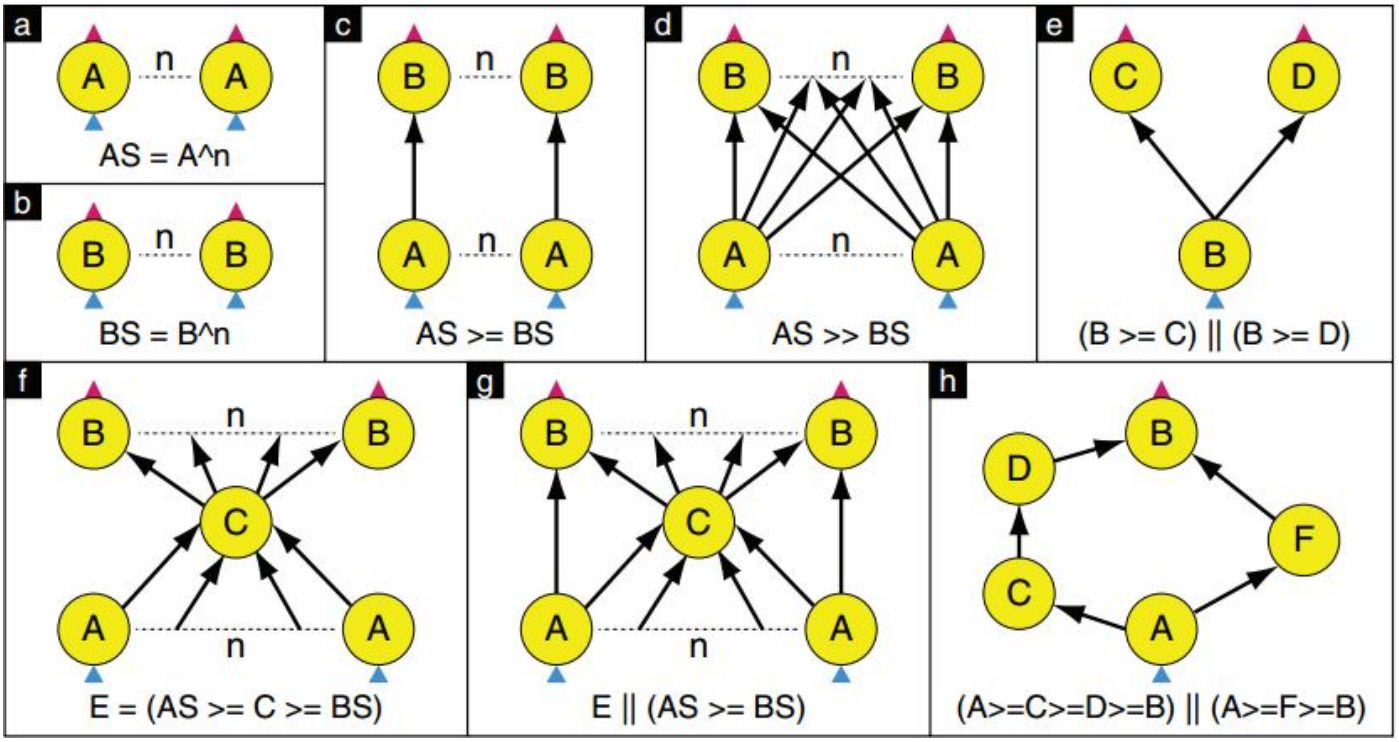


Fig. 3. The operators of the graph description language Source:[6]

time based on the output of the computations. However, there are some restrictions that do not allow to delete a vertex once it has been executed or alter the number and type of channels [5].

5. USE CASES

Dryad can be used for any large scale computational applications, Scientific applications, Large-data processing applications-indexing, search,etc. High-level language compilers use Dryad as a run-time. For example, SCOPE (Structured Computations Optimized for Parallel Execution) and DryadLINQ [7]. Professor of informatics at Indiana University, Geoffrey Fox had used Dryad and DryadLINQ to analyze RADAR data focused on glaciers to earn more about the earth's past and its present in order to make more informed, potentially life-saving predictions about its future [8].

6. PERFORMANCE

Computers	1	2	3	4	5	6	7	8	9
SQLServer	3780								
Two-pass	2370	1260	836	662	523	463	423	346	321
In-memory						217	203	183	168

Fig. 4. Time taken to process SQL query Source:[9]

The performance of Dryad was experimentally evaluated through two processes. In the first a relatively simple SQL query was distributed among 10 computers in the Dryad system and it performance was compared with that of a traditional commercial SQL server. The second is a map-reduce data-mining operation applied to 10.2 TBytes of data using a cluster of around 1800 computers [9]. The experiments were run in Microsoft Research laboratory with computers having 2 dual-core Opteron processors running at 2 GHz, 8 GB of DRAM and 4 disks(400 GB). Network connectivity was by 1 Gbit/sec Ethernet links connecting into a single non-blocking switch. Figure 4 summarizes the time in seconds to process an SQL query using different numbers of computers [9].

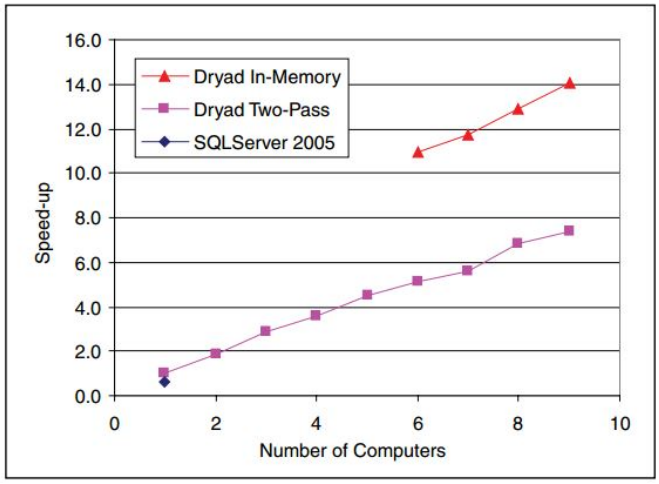


Fig. 5. speedup of the SQL query Source:[9]

From Figure 5 we can see that the speedup of the SQL query computation is nearly linear in the number of computers used. The baseline is relative to Dryad running on a single computer and times are as given in Figure 4 [9].

7. CONCLUSION

Microsoft Dryad are suitable only for applications that take the form of a directed acyclic graph. Dryad assumes that the vertices are deterministic and will fail if the application contains non-deterministic vertices. Since the job manager assumes that it has exclusive control over the computers in the cluster, it is difficult to efficiently run more than one job at a time Dryad [9]. It focuses only on throughput and does not improve the latency. In November 2012, Microsoft shifted their focus to Hadoop rather than improvising.

REFERENCES

[1] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in *ACM SIGOPS operating systems review*, vol. 41, no. 3. Lisboa-Portugal: ACM, Mar. 2007, p. 59, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2007/03/eurosys07.pdf

[2] C. Poulain, "Dryad - microsoft research," Web Page, Mar. 2007, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/project/dryad/

[3] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in *ACM SIGOPS operating systems review*, vol. 41, no. 3. Lisboa-Portugal: ACM, Mar. 2007, pp. 59–72, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2007/03/eurosys07.pdf

[4] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in *ACM SIGOPS operating systems review*, vol. 41, no. 3. Lisboa-Portugal: ACM, Mar. 2007, pp. 60–61, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2007/03/eurosys07.pdf

[5] M. Isard, "Dryad - google tech talks," Web Page, Nov. 2007, accessed: 2017-2-24. [Online]. Available: https://www.youtube.com/watch?v=WPhE5JCP2Ak

[6] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in *ACM SIGOPS operating systems review*, vol. 41, no. 3. Lisboa-Portugal: ACM, Mar. 2007, pp. 63–64, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2007/03/eurosys07.pdf

[7] Wikipedia, "Dryad (programming)-wikipedia," Web Page, Nov. 2016, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/blog/dryad-and-dryadlinq-academic-accelerators-for-parallel-data-analysis/

[8] D. Campbell, "Dryad and dryadlinq: Academic accelerators for parallel data analysis," Web Page, Feb. 2010, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/blog/dryad-and-dryadlinq-academic-accelerators-for-parallel-data-analysis/

[9] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in *ACM SIGOPS operating systems review*, vol. 41, no. 3. Lisboa-Portugal: ACM, Mar. 2007, pp. 71–72, accessed: 2017-2-24. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2007/03/eurosys07.pdf

A Report on Apache Apex

SRIKANTH RAMANAM[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: srikrama@iu.edu*

April 30, 2017

Apache Apex is a Hadoop YARN native big data processing platform with both stream and batch processing capabilities. This paper explores the architecture, functioning and competition of Apache Apex.

Keywords: Stream, Processing, YARN, Apache, Apex, Malhar, I524

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IR-2028/report.pdf

1. INTRODUCTION

Apex is an enterprise-grade stream and batch processing platform for the Apache Hadoop ecosystem [1]. It was initially developed by DataTorrent as the core engine for RTS,a data processing and analytics platform. Apex was submitted to Apache incubator in 2015 [2]. Later several enterprises like CapitalOne, DirecTV, General Electric, Apple and Silver Spring Networks joined its open source community. Apache Apex was first released in 2016.

Apache Apex is built over YARN and is compatible with existing Hadoop platforms, allowing users to leverage their previous Hadoop investments and applications [3]. According to Apache Apex website [4], " it processes big data in-motion in a way that is highly scalable, highly performant, fault tolerant, stateful, secure, distributed, and easily operable". Apex automatically handles operational aspects like state management, fault tolerance, scalabilty etc [5]. It also provides a simple API that supports Java, facilitating easy development and widespread adoption [5]. It also has a REST API facilitating compatibility with popular web technologies. Apache Apex also has a metrics API that allows users to monitor various aspects of operators in real time.

2. COMPONENTS

Apache Apex has two main components. They are Apex Core and Apex Malhar [6].

2.1. Apex Core

Apex Core is the framework for building distributed stream processing and analytics applications on Hadoop. It also enables building of unified batch and stream processing applications.

2.2. Apex Malhar

Malhar provides a library of operators that perform widely used functionality. These reusable blocks reduce the amount of coding required to build applications and enable users to speed up application development. Operators offered by are mainly of two types

2.2.1. *Input/Output Operators*

Input/Output operators: These operators offer connectivity with a variety of existing data sources.

2.2.2. *Compute Operators*

Compute Operators: These operators offer functionality of Machine Learning, Stats and Math, Pattern Marching, Query and Scripting, Stream manipulators, Parsers and UI & Charting.

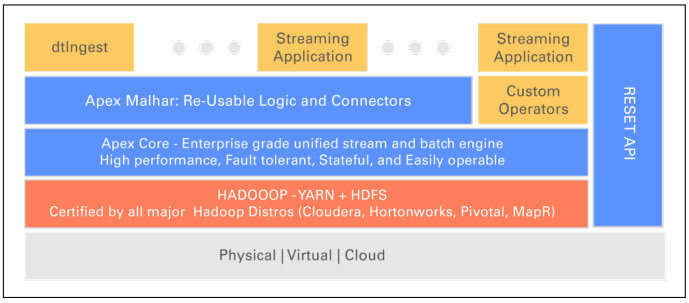


Fig. 1. Apache Apex Components [3]

3. ARCHITECTURE

Operators are the basic blocks of Apex applications. A streaming application is built using in-built or custom operators are connected to form a DAG (Directed Acyclic Graph) using streams.

4. APPLICATION DEVELOPMENT

Apex applications can be written in Java using any IDE supporting Java like Eclipse. Other prerequisites include Apache Maven

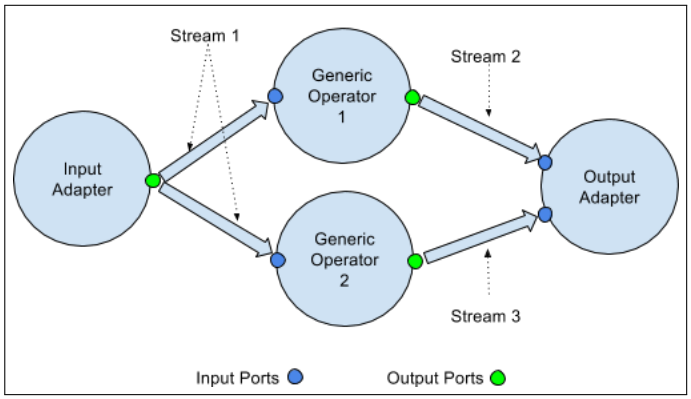


Fig. 2. Apex Application DAG [7]

3.0., Apache Apex, Apache Malhar [7].

4.1. Operators

Operators are independent units of logical operations that either contribute to a part of or a whole business use case. An operator has an input port to receive data tuples and an output port to send data tuples to another operator or external system [8].

4.1.1. Types of Operators [8]

- Input Adapter: An operator at the beginning of the DAG to receive data from an external system.
- Generic Operator: Accepts tuples from previous operator in DAG and does some processing task and outputs the processed data to another operator.
- An operator at the end of a DAG and outputs the data tuples to an external system.

4.1.2. Operator API [8]

- setup() initializes the operator.
- process() performs the core processing operations on data tuples and gets triggered when tuples are received.
- beginWindow() and endWindow() are used for pre and post processing steps.
- teardown() shuts down the operator and releases the resources held by the operator.

4.2. Directed Acyclic Graph

A Directed Acyclic Graph(DAG), is constructed to accomplish a business task using several operators connected through streams [7]. A stream is a sequence of data tuples. To construct a DAG, operators are added using dag.addOperator(args) while streams are added using dag.addStream(args). Other configurations related to YARN can also be added to DAG [9].

4.3. Package

Apex applications are assembled and shared using Apache Apex Packages, which are zip files with all necessary files to launch those applications. Apache Apex Packages are created using Maven. First a Maven project is created with path to the application code. Then a mvn package command creates an application package in the target package. Zip structure of a mvn package consists of app with jar files of the DAG code, lib with jar files of dependencies, conf with preset configurations, META-INF consisting of meta information in files and resources for other files [10].

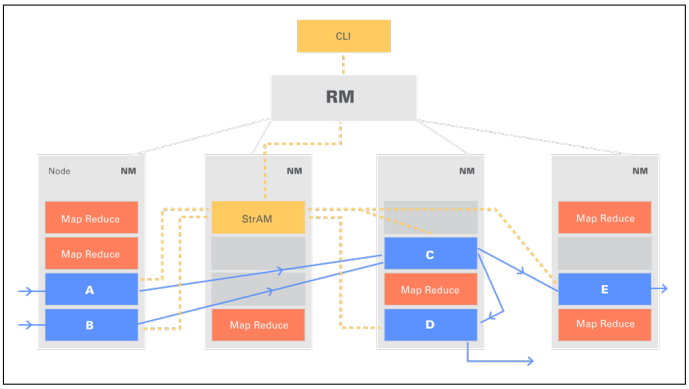


Fig. 3. Apache Apex Application Example [3]

5. FUNCTIONING

Apache Apex applications are built as DAGs consisting of operators and streams. These applications are packaged, shared and deployed over Hadoop clusters.

5.1. Fault tolerance and Recovery

The states of operators and application master are checkpointed regularly to a persistent store like HDFS [11]. Failed operators are automatically detected through continuous monitoring [11]. When a failure occurs, the checkpointed states of operators are used to revive the application [11]. Data can be made to replay from the checkpointed state of the operator after recovery preventing data loss [11].

5.2. Compatibility

Apache Apex is also compatible with several popular data file systems, message systems and database systems through connectors provided by Malhar. They are shown in the below figure.

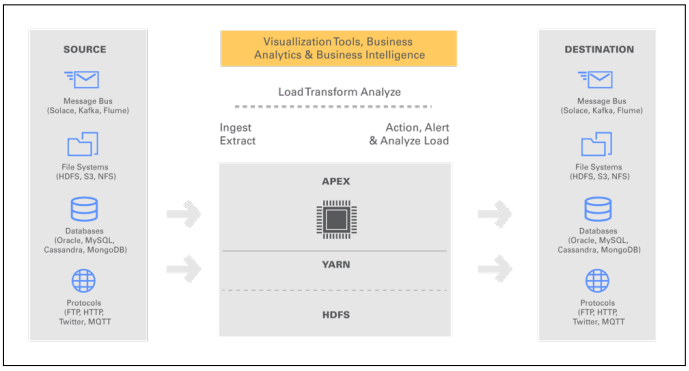


Fig. 4. Apache Apex Interoperability[3]

6. USER INTERFACE

A command line interface called Apex CLI is available for Apache Apex [12]. It can be launched with command apex. help

command is available for all commands to obtain information and syntax.

Apache Apex's REST, Java API's can be integrated with commonly used existing web technologies to create web interfaces for visual analytics. Data Torrent also offers RTS platform, built on Apex, which provides visualization with several real-time dashboards that help monitor streaming applications [11].

7. LICENSE AND PRICING

Apache Apex is a free and open source software. It is licensed under Apache License 2.0 [4].

8. COMPETITION

Some of the competitors of Apache Apex for stream processing and analytics are [13]:

8.1. Apache Spark

This is a large-scale data processing engine that also offers stream processing [14]. But this is not a pure streaming engine as it accomplishes the same through micro-batching, fast execution of batches on small sets of data.

8.2. Apache Flink

Apache Flink is an open-source stream processing framework that processes streams in real-time [15]. This is almost similar to Apex but not as widely used.

8.3. Apache Storm

This is a free and open source distributed real-time computation system [16]. This is fast but not stateful like Apache Apex.

8.4. Apache Samza

Apache Samza is a distributed stream processing framework [17].It was first developed by LinkedIn and later opensourced [18]. It is built on top of Apache Kafka [19], a distributed streaming platform. It provides stateful streaming capabilities [18]. It has great compatibility with Kafka [18]. Streaming applications can be built in such that Kafka consumes the data processed by Samza [18].

9. USERS

With its stream processing capabilities, Apache Apex facilitates building large scale real-time analytics applications. Enterprises like GE, PubMatic, SilverSpring Networks are using Apex based streaming solutions [9].

10. CONCLUSION

Apache Apex is an open source YARN(Hadoop 2.0)-native platform [6]. It unifies stream and batch processing. It can be used for processing both streams of data and static files making it more relevant in the context of present day internet and social media. It is aimed at leveraging the present Hadoop platform and reducing the learning curve for development of applications over it. It is aimed at It can used through a simple API. It enables reuse of code by not having to make drastic changes to the applications by providing interoperability with existing technology stack. It leverages the existing Hadoop platform investments.

ACKNOWLEDGEMENTS

This paper has been written as part of a class assignment for the course: I524: Big Data Software and projects, Spring 2017, School of Informatics and computing, Indiana University, Bloomington. Special thanks to Professor Gregor von Laszewski, Dimitar Nikolov and all associate instructors for guiding through the process of writing this paper.

REFERENCES

[1] Apache, "Apache software foundation blog apex introduction," Web Page, Apr. 2015, accessed: 2017-03-26. [Online]. Available: https://blogs.apache.org/foundation/entry/the_apache_software_foundation_announces90

[2] A. Kekre, "Apache apex blog incubator," Web Page, Sep. 2015, accessed: 2017-03-26. [Online]. Available: https://www.datatorrent.com/blog/apex-accepted-as-apache-incubator-project/

[3] A. Kekre, "Apache apex blog introduction," Web Page, Sep. 2015, accessed: 2017-03-26. [Online]. Available: https://www.datatorrent.com/blog/introducing-apache-apex-incubating/

[4] Apache, "Apache apex," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/

[5] Apache, "Apache apex documentation," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/docs/apex/application_packages/#apache-apex-packages

[6] Wikipedia, "Apache apex wiki," Web Page, accessed: 2017-03-26. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Apex

[7] Apache, "Apache apex application development documentation," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/docs/apex/application_development/

[8] Apache, "Apache apex application operator documentation," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/docs/apex/operator_development/

[9] T. Weise, "Apache apex slideshare," Web Page, Jul. 2016, accessed: 2017-03-26. [Online]. Available: https://www.slideshare.net/ThomasWeise/apache-apex-stream-processing-architecture-and-applications

[10] Apache, "Apache apex application package documentation," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/docs/apex/application_packages/#apache-apex-packages

[11] Apache, "Apache apex introduction slideshare," Web Page, Jul. 2016, accessed: 2017-03-26. [Online]. Available: https://www.slideshare.net/ThomasWeise/apache-apex-stream-processing-architecture-and-applications

[12] Apache, "Apache apex cli documentation," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/docs/apex/operator_development/

[13] S. Hall, "The newstack article on apache apex competition," Web Page, May 2016, accessed: 2017-03-26. [Online]. Available: https://thenewstack.io/apache-gets-another-real-time-stream-processing-framework-apex/

[14] Apache, "Apache spark," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: http://spark.apache.org/

[15] S. Hall, "The newstack article on apache flink," Web Page, Apr. 2016, accessed: 2017-03-26. [Online]. Available: https://thenewstack.io/apache-flink-addresses-continuous-stream-processing/

[16] Apache, "Apache storm," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: http://storm.apache.org/

[17] Apache, "Apache samza," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: http://samza.apache.org/

[18] S. Hall, "The newstack article on apache streaming projects," Web Page, Jun. 2016, accessed: 2017-03-26. [Online]. Available: https://thenewstack.io/apache-gets-another-real-time-stream-processing-framework-apex/

[19] Apache, "Apache kafka," Web Page, 2016, accessed: 2017-04-06. [Online]. Available: https://kafka.apache.org/

Apache Mahout

NAVEENKUMAR RAMARAJU[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors:naveenkumar2703@gmail.com*

Paper 2, April 30, 2017

Apache Mahout[1] is an extensible programming environment to build scalable machine learning algorithms. It has algorithms to work in tandem with frameworks like Hadoop, Spark, Flink and H2O which specializes on dealing with large scale data. Samsara is a vector math environment to do linear algebra operations using distributed computing.

Keywords: Mahout, Samsara, Apache, Scalable, Machine learning

https://github.com/naveenkumar2703/sp17-i524/paper2/S17-IR-2029/report.pdf

1. INTRODUCTION

Mahout is an open source software to create scalable machine learning models. The initial version of Mahout targeted to implement all ten machine learning algorithms discussed in Andrew Ng's paper "Map-Reduce for Machine Learning on Multicore"[2] with scalability in mid. Further releases of Mahout added various implementations of clustering, classification, collaborative filtering and genetic algorithms in Java for usage in single machine as well as in clusters using map reduce.

As the popularity of in-memory softwares like Spark started gaining popularity over disk based softwares like Hadoop, new features rolled out by Mahout did not support MapReduce. "Samsara" is a module to do algebraic operations which was released in Mahout version 0.10 and is compatible only with frameworks like Spark[3], H2O[4] and Flink[5] but not MapReduce based frameworks. Samsara was mainly written in Scala and is optimized to operate well independent of the background. Another key feature of Samsara is that it supports R-like syntax for linear algebra operations.

Mahout has many algorithms for MapReduce based frameworks like Hadoop. Only some of them are implemented for in-memory based frameworks like Spark. A full list of algorithms available in Mahout and the frameworks supported by them is discussed in section 2.

2. FEATURES

Mahout supports wide range of machine learning algorithms like classification, collaborative filtering and clustering, dimensionality reduction techniques like SVD, PCA and QR decomposition[6]. Mahout Samsara environment provides linear algebra and statistics operations. Each of these are described in this section.

2.1. Samsara - Math Environment

Mahout Samsara[7] is a math environment to create and perform various math and linear algebra operations. Some of the key functionalities are BLAS (Basic Linear Algebra Subprogram), distributed row matrix, distributed ALS (Alternating Least Squares), PCA (principal component analysis), incore and distributed SPCA (Stochastic PCA), SVD (singular value decomposition), incore and distributed SSVD (Stochastic SVD), Eigen decomposition, Cholesky decomposition and similarity analysis.

One of the main advantage of Samsara is it supports R and Matlab like syntax using Scala's DSL (domain specific language) feature. DSL's are syntactic sugars for easy interpretation. An example is provided in section 5. Mahout Samsara is supported on Spark, H2O and Flink engines. Samsara is not available in Hadoop and MapReduce based engines but has a different implementation that supports all these math operations.

2.2. Classification

Mahout has logistic regression using stochastic gradient descent, naive Bayes, complementary naive Bayes, random forest and hidden markov algorithms for classification. Naive Bayes available in Spark and MapReduce is the only distributed classification algorithm. Others are supported only on single machines.

2.3. Clustering

Mahout supports clustering algorithms like K-means, fuzzy k-means, streaming k-means, spectral clustering and canopy clustering. These are available only for single machines and MapReduce based environments.

2.4. Collaborative Filtering

Mahout has implementation for user based and item based collaborative filtering algorithms for single machine, MapReduce

and spark engines. It also has implementation of matrix factorization with ALS, weighted matrix factorization using SVD for single machine and MapReduce.

2.5. Other

Mahout supports several other features like Latent Dirichlet Allocation, row similarity job, collocations, sparse TF-IDF (Term frequency - inverse document frequency, a common feature used in information retrival and document search) vectors from text, XML Parsing, Email Archive Parsing for MapReduce engines. It also has Evolutionary Processes/Genetic algorithms implementation that runs on single machine.

3. LICENSING

Apache Mahout is an open source software available for free commercial usage and is licensed under Apache License, Version 2.0[8]. Associated frameworks like Spark, Hadoop, Flink and H2O are also open source products.

4. ECOSYSTEM

Mahout can be used with wide variety of distributed systems like Spark, H2O, Flink and Hadoop. It can also be used on single machine. A key benefit of Mahout is that the machine learning or math code can be used and written in same syntax independent of backends.

An illustration of how Mahout works by using Scala DSL for math operations with various engines is illustrated in figure 1.

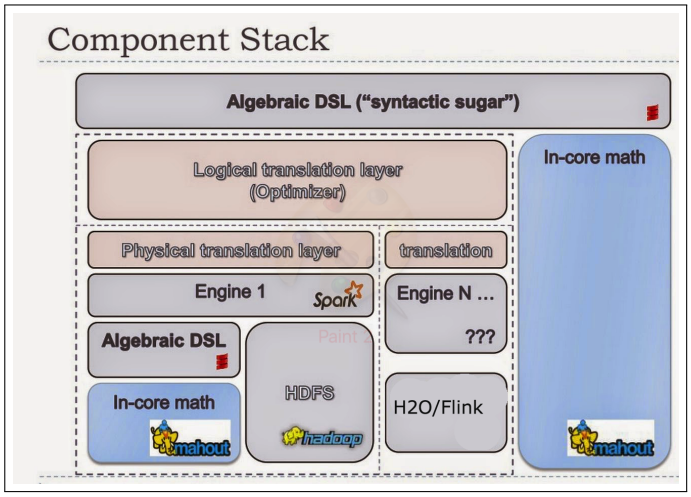


Fig. 1. Mahout Ecosystem.
Source: [9]

5. USE CASES

An use case for recommendation system and a simple linear algebra operation is provided in this section.

5.1. Recommendation system with Spark Engine

A recommendation system is used to recommend most relevant product that he/she might be interested and boost the sales in online platforms. Recommendation can be done based on user similarity or item similarity. In case of user similarity, recommendation is provided based on idea that users with similar interest would buy similar products. Similar users are identified, grouped and recommendations are made based on difference in set of products. Where as item similarity is based on identifying the products that were bought together and recommending the products that is not in customer's basket or purchase history. Recommendation based on item similarity is very popular in industry as number of product types is way less than number of customers with regular purchase patterns in large retail industry.

An illustration of how Mahout can be used with Spark for item based recommendation is illustrated in figure 2.

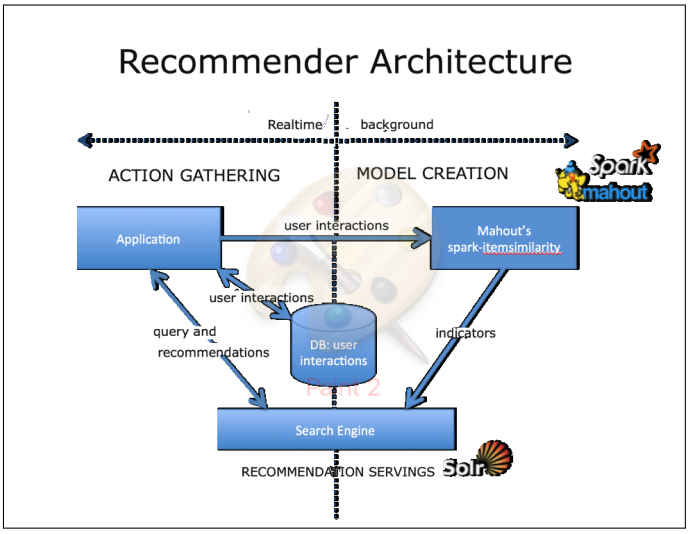


Fig. 2. Recommender Illustration.
Source: [10]

Mahout's MapReduce version of item similarity takes a text file that is expected to have user and item IDs and provide recommendations based on content.

5.2. SPCA

The equation to compute stochastic principal component analysis is given in equation 1

$$G = BB' - C - C' + s_q s_q' \xi' \xi \quad (1)$$

where G is the matrix representation of the result of SPCA, B is the input or feature matrix, C is correlation matrix of the inputs with B′ and C′ as their transpose matrix, s_q is standard deviation of each feature and ξ is covariance matrix and s_q', ξ' are their respective transpose matrices.

One line Mahout code to compute this using Scala DSL is illustrated here,

val g = bt.t %*% bt - c - c.t + $(s_q crosss_q) * (xi\ dot\ xi)$

This could be used in single or distributed machine in Spark, H2O or Flink to perform SPCA.

6. USEFUL RESOURCES

Apache Mahout Cookbook[11] by Piero Giacomelli is a good introductory book on Mahout. Apache Mahout Beyond MapReduce[12] by Dmitriy Lyubimov provides an exhaustive coverage of Mahout Samsara and math operations.

Mahout website[13] has a compilation of useful resources like books, tutorials and talks about Mahout and machine learning.

7. CONCLUSION

Apache Mahout provides an open source environment with scalable algorithms for machine learning and math operations using DSL. It can be used with multiple backends like Spark, Scala, Flink and Hadoop. Same code can be used for single machine and distributed computing of algorithms in any supported backends.

ACKNOWLEDGEMENTS

This work was done as part of the course "I524: Big Data and Open Source Software Projects" at Indiana University during Spring 2017. Thanks to our Professor Gregor von Laszewski and associate instructors for their help and support during the course.

REFERENCES

[1] Apache Software Foundation, “Apache mahout,” Web Page, Jun. 2016, accessed: 2017-3-26. [Online]. Available: http://mahout.apache.org/

[2] C. tao Chu, S. K. Kim, Y. an Lin, Y. Yu, G. Bradski, K. Olukotun, and A. Y. Ng, “Map-reduce for machine learning on multicore,” in *Advances in Neural Information Processing Systems 19*, P. B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. MIT Press, 2007, pp. 281–288, accessed: 2017-3-26. [Online]. Available: http://papers.nips.cc/paper/3150-map-reduce-for-machine-learning-on-multicore.pdf

[3] Apache Spark Community, “Apache spark,” Web Page, Feb. 2017, accessed: 2017-4-4. [Online]. Available: https://spark.apache.org/

[4] H20.ai, “H20,” Web Page, Jan. 2016, accessed: 2017-4-4. [Online]. Available: https://www.h2o.ai/h2o/why-h2o/

[5] Apache Flink Community, “Introduction to Apache Flink,” Web Page, Mar. 2016, accessed: 2017-4-4. [Online]. Available: https://flink.apache.org/introduction.html

[6] Apache Software Foundation, “Mahout Features by Engine,” Web Page, Jan. 2016, accessed: 2017-4-4. [Online]. Available: https://mahout.apache.org/users/basics/algorithms.html

[7] Apache Software Foundation, “Mahout samsara incore references,” Web Page, May 2016, accessed: 2017-3-26. [Online]. Available: http://mahout.apache.org/users/environment/in-core-reference.html

[8] Apache Software Foundation, “Apache License 2.0,” Web Page, Jan. 2004, accessed: 2017-3-26. [Online]. Available: http://www.apache.org/licenses/LICENSE-2.0

[9] D. Lyubimov and A. Palumbo, “Mahout 0.10.x: first mahout release as a programming environment,” Web Page, Apr. 2015, accessed: 2017-3-26. [Online]. Available: http://www.weatheringthroughtechdays.com/2015/04/mahout-010x-first-mahout-release-as.html

[10] Apache Software Foundation, “Intro to cooccurrence recommenders with spark,” Web Page, May 2016, accessed: 2017-3-26. [Online]. Available: http://mahout.apache.org/users/algorithms/intro-cooccurrence-spark.html

[11] P. Giacomelli, *Apache Mahout Cookbook*. Packt Publishing, 2016, accessed: 2017-3-26.

[12] D. Lyubimov and A. Palumbo, *Apache Mahout: Beyond MapReduce*. Createspace Independent Publishing Platform, 2016, accessed: 2017-3-26.

[13] Apache Software Foundation, “Mahout book, tutorials, talks,” Web Page, May 2016, accessed: 2017-3-26. [Online]. Available: https://mahout.apache.org/general/books-tutorials-and-talks.html

Neo4J

SOWMYA RAVI[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: sowravi@iu.edu.com*

project-000, April 30, 2017

Neo4J is a graph database designed for fast data access and management. The data is stored in the form of nodes and relationships in Neo4J. The unique approach it takes to store data makes it an efficientent to store data that contain a large the number of relationships. Moreover, it has the ability to store trillions of data entries in a compact manner. Neo4J comes along with Cypher, a highly readable querying language. Neo4j achieves the high efficiency and throughput by distributed computing. The various modes of clustering in Neo4j renders the capability of distributed computing. This paper focuses elaborates on the architecture of Neo4j and its uses [1].

Keywords: Cloud, I524

https://github.com/cloudmesh/classes/blob/master/docs/source/format/report/report.pdf

1. INTRODUCTION

Certain problems present in the world cannot be solved by using relational databases. For e.g. a social graph representing the network of friends in a social networking website. In this case the number of relationships in the data is too extensive and the relational databases perform poorly. Graph data bases on the other hand make the task of storing large amounts of data relatively simple and efficient. Neo4J is one such NoSQL, graph database which was developed to be used in the kind of problems mentioned before [2].

Neo4j is an open source data management software. At its core, Neo4j stores data in the form of nodes and relationships. It is often deployed in a production environment as a fault tolerant cluster of machines. The high scalability and fast traversal times make it far more efficient than the conventional relational databases [1].

2. CYPHER PROGRAMING LANGUAGE

Neo4j uses its own programming language, Cypher, for data creation as well as querying. Cypher is capable of doing SQL like actions. In addition, it can specifically perform a powerful query called traversals. Traversal involves moving along a specific set of nodes in the database thereby tracing a path. This allows to leverage the spatial structuring of the data to get valuable information, similar to network analysis [3].

3. CLUSTERING

This section discusses Neo4js architecture with respect to clustering. Clustering is the process of grouing instances or items. In the context of Neo4j, a number of maches are clustered in a group. The machines are connected and can communicate with one another. Each machine can be visulaized as a separate node in the cluster which performs a part or whole of the task assigned by a process to Neo4j. Neo4j uses clustering of machines to achieve high throughput, availability and disaster recovery [4]. Neo4j offers two kind of clustering

1. Causal Clustering
2. Highly Available clustering

Each type of cluster has a unique set of features. The mode of clustering is chosen according to the application and requirements.

3.1. Causal Clustering

The Causal clustering of machines in Neo4j is aimed at providing two important features: safety and scalability. [5]

For operational purposes, the cluster is usually separated into two components: the core servers and the read replicas. The architecture of causal clustering is shown in Fig.2. All the write operations coming from the app servers are performed by the core servers while data is read by the app servers from the read replicas. The following sub-sections datail the wrorking of core servers and read replicas and also how they ensure safe and scalable data storage.

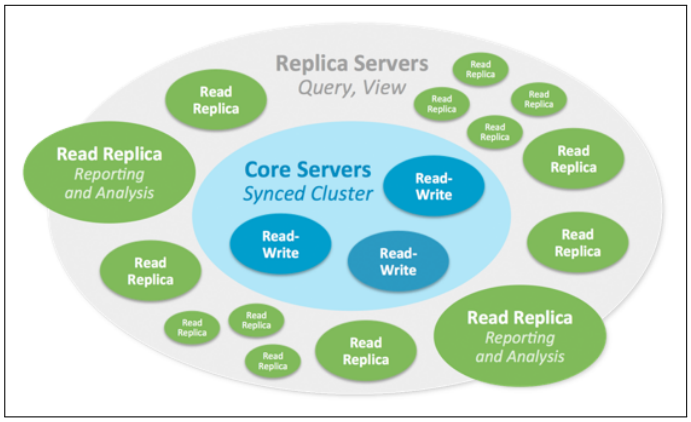


Fig. 1. Architecture of Causal Clustering [5]

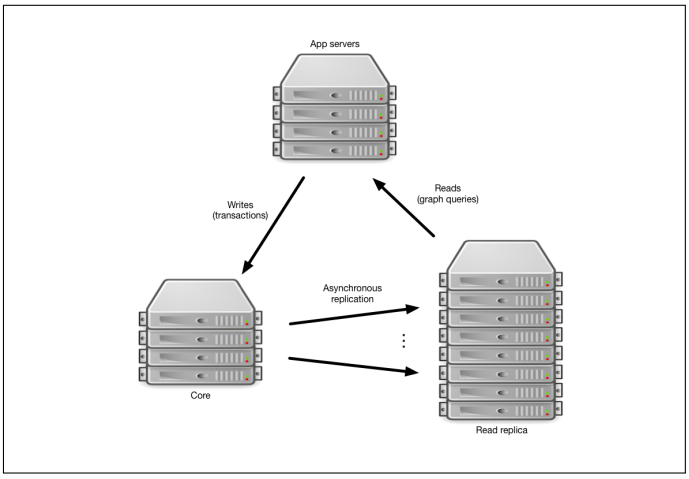


Fig. 2. Causal Cluster with causal consistency set via Neo4j drivers [5]

3.1.1. Core Servers

. The Core servers are responsible for safe data storage. This is achieved by replicating all incoming queries/transactions using Raft protocol (A log replication protocol) [5]. The protocol ensures the durability of data before committing to the query request. Usually, a transaction is accepted only when a majority of the servers,calculated as $N + 1/2$, have accepted it. This number is directly proportional to the number of core servers N. Hence, as the number of core servers grows, the size of majority required for committing to an end user also increases [5].

In practice, few machines in the core server cluster is enough to provide fault tolerance. This number is calculated using the formula: $N = 2F + 1$ where N is the number of servers required to tolerate F faults [5]. When a core server suffers a large number of faults, it is automatically converted to a read-only server for safety purposes.

3.1.2. Read replicas

Read Replicas are Neo4j databases that scale out the incoming queries and procedures. They act like cache memories to the core servers which safeguard the data. Even though the read replicas are full-fledged databases, they are equipped to perform arbitrary read-only activities [5].

Read Replicas are created asynchronously by core servers through log-shipping [5]. Log shipping occurs when the read replicas poll the core servers for new transactions and the transactions are shipped from the core severs to the read replicas. This polling occurs periodically. Usually, a small number of core servers ship out queries to a relatively large number of read replicas, allowing a large fan out of workload thereby, achieving scalability [5]. The read replicas unlike the core servers do not participate in deciding the cluster topology.

3.1.3. Causal Consistency

In applications, data is generally read from a graph and written to a graph. In order to ensure the causal consistency in the data, the write operation must take into account previous write operations. The Causal Consistency model for distributed computing requires every node in the system to see causally related operations in the same order. This model ensures that the data can be written to cores and the written data be read from read replicas. Fig.3 illustrates a Causal Cluster with causal consistency

4. HIGHLY AVAILABLE CLUSTER

The causal cluster discussed in the previous section focuses on safety and security. There is a division of work between the core servers and read replicas in the causal cluster. The Highly available cluster however, ensures continuous availablity. In this type of cluster each instance of the cluster contains full copy of the data in their local database. Thus each instance in the cluster is fully capable of performing all operations thereby achieving high availability. The cluster can be visualized as containing a single master with multiple slaves in which each instance is connected to every other instance(A 3 member cluster is shown in Fig.4)

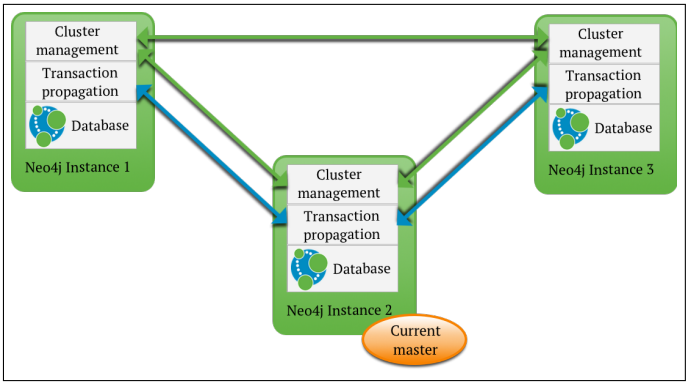


Fig. 3. A Highly Available cluster model [6]

Also, each instance contains the logic to perform read/write operations and election management [6]. Every slave, excluding the Arbiter instance periodically communicate with the master to keep databases up to date [6]. There is a special slave called the Arbiter explained in the following section.

4.0.1. Arbiter Instance

The Arbiter instance is a special slave that participates in cluster management activities but does not contain any replicated data. It simply contains a Neo4j software running in arbiter mode [6].

4.0.2. Transaction Propagation

Write Transactions performed directly on the Master will be pushed to slaves once the transaction is successful. When a write transaction is performed on a slave, the slave synchronizes with the Master after each write operation. The write operation on slave is always performed after ensuring that the slave is synchronized with the Master [6].

5. USE CASES

Neo4j can be used for fraud detection, network and IT operations, Real-time recommendation system, Social Network and Identity and Access Management [7]. A sample use case in fraud detection is described below.

5.0.1. Neo4j for First Party Fraud Detection

First party fraud occurs when a person uses illegitimate information to secure credit card. Often two or more people share certain information to create multiple identities and operate as a ring. Neo4j is a great tool to detect fraud rings. In Neo4j, all the customer related data is stored as a directed graph. When more than two people share the same contact information like address or SSN and have outstanding loans or credit, this may potentially be a fraud ring. It is possible to discover similar connections by writing simple queries in Cypher. In relational Databases, this network is present in tables with rows and columns. Complex joins have to performed to find a fraud ring and joins are expensive operations. Moreover, in real-time analysis scalability could be a serious issue. Neo4j achieves scalability when operated in a causal clustering mode. The simplicity and robustness of Neo4j makes it a great tool for fraud detection [8].

6. CONCLUSION

Neo4j being an open source, graph based and a highly scalable software, it is suitable for applications that deal with data containing numerous relationships. It offers two different clustering modes it can operate in, each focusing on a certain feature. In addition, Cypher the query language is very simple and finds relationships without performing costly operations like joins. The simplicity and versatility of Neo4j makes it a great software aid for data scientists trying to analyze relationships and networks in real time as well as in batch.

REFERENCES

[1] Neo4j, "Chapter 1. introduction," Web Page, last Accessed: 03.24.2017. [Online]. Available: https://neo4j.com/docs/operations-manual/current/introduction/

[2] S. Haines, "Introduction to neo4j," Web Page, last Accessed: 03.24.2017. [Online]. Available: http://www.informit.com/articles/article.aspx?p=2415370

[3] R. Ostman, "Graphical database of citation network analysis," PDF Document, last Accessed: 03.24.2017. [Online]. Available: http://webdocs.ischool.illinois.edu/crt/ostman.pdf

[4] Neo4j, "Chapter 4. clustering," Web page, last Accessed: 2017.02.24. [Online]. Available: https://neo4j.com/docs/operations-manual/current/clustering/

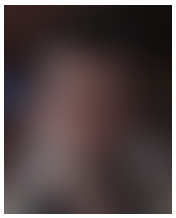
[5] Neo4j, "Causal cluster," Web page, last Accessed: 2017.03.24. [Online]. Available: https://neo4j.com/docs/operations-manual/current/clustering/causal-clustering/introduction/

[6] Neo4j, "Highly available cluster," Web page, last Accessed: 2017.03.24. [Online]. Available: https://neo4j.com/docs/operations-manual/current/clustering/high-availability/architecture/

[7] Neo4j, "Graph database use cases," Web page, last Accessed: 2017.03.24. [Online]. Available: https://neo4j.com/use-cases/

[8] P. Sadowksi, Gorka Rathle, *Fraud Detection: Discovering Connections with Graph Databases*. Ne04j, 2015, last Accessed: 04.08.2017.

AUTHOR BIOGRAPHIES



Sowmya Ravi pursuing Masters in Data Science from Indiana University. Her research interests include Machine Learning, Data Mining and Big Data Analytics

OpenStack Nova: Compute Service of OpenStack Cloud

KUMAR SATYAM[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: ksatyam@indiana.edu*

paper-2, April 30, 2017

OpenStack Nova is the compute service of the OpenStack cloud system.It is designed to manage and automate the pools of computer resources and can work on bare metal and high performance computing. It is written in python.We will discuss the main components included in the Nova Architecture [1].

Keywords: Cloud,OpenStack,Nova, API, I524

https://github.com/satyamsah/sp17-i524/blob/master/paper2/S17-IR-2031/report.pdf

1. INTRODUCTION

Nova is responsible for spawning vm instances in openstack environment. It is built on a messaging architecture which runs on several servers. This architecture allows components to communicate through a messaging queue. Nova together shares a centralized SQL-based database for smaller deployment. For large deployments an aggregation is used to manage data across multiple data stores[2].

3. NOVA FRAMEWORK

Nova is comprised of multiple server processes, each performing different functions. The OpenStack provided user interface for Nova which is a REST API. During invocation of the API, the Nova communicates via RPC (Remote procedure call) passing mechanism.As shown in Fig.1, it interacts with other OpenStack components like Keystone , Glance, Cinder etc.

The API servers process REST requests, which typically involve database reads/writes. RPC messaging is done via the 'oslo.message' library. Most of the nova components can run on different servers and have a manager that is listening for RPC messages. One of the components is Nova Compute where a single process runs on the hypervisor it is managing.

Nova has a centralized database that is logically shared between all components[4].

4. NOVA COMPONENTS

Below are the major components of Nova:

DB: An SQL database for data storage.This is the SQLAlchemy-compatible database. The database name is

2. ARCHITECTURE -OPENSTACK NOVA

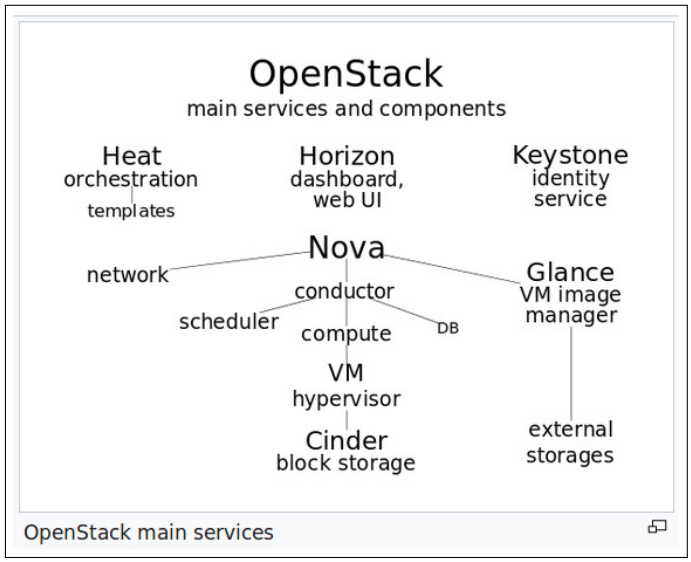


Fig. 1. Architecture of OpenStack cloud using Nova [3]

'nova'. The 'nova-conductor' service is the only service that writes to the database.

API: It is the component that receive HTTP requests, converts commands and communicates with other components via the 'oslo.messeging' queue or HTTP. The 'python-novaclient 7.1.0' is a python binding to OpenStack Nova API which acts as a client for the same. The nova-api provides an endpoint for all API queries (either OpenStack API or EC2 API), initiates most of the orchestration activities (such as running an instance) and also enforces some policy (mostly quota checks)

Schedular: The 'nova-schedular' is a service to determine how to dispatch compute requests.For example, it determines on which host a VM should launch. Compute is configured with a default scheduler options in the /etc/nova/nova.conf file[5].

Network: It manages IP forwarding, bridges, and vlans. The network controller with nova-network provides virtual networks to enable compute servers to interact with each other and with public network. Compute with nova-network support the following modes, which are implemented as Network Manager types:

- Flat Network Manager
- Flat DHCP Network Manager
- VLAN Network Manger

volume : It manages creation, attaching and detaching of persistent volumes to compute instances. This functionality is being migrated to Cinder, a separate OpenStack service

Compute: It manages communication with hypervisor and virtual machine

Conductor: It handles requests that need coordination, acts as a database proxy, or handles object conversions

5. MAJOR TASK PERFORMED BY OPENSTACK NOVA SERVICE

Nova service performs some major task.[6].Some of them are authenticating against the nova-api endpoint, listing instances, retrieving an instance by name and shutting it down, booting an instance and checking status, attaching Floating IP address,changing a security group, and Retrieving console login

6. NOVA AND OTHER OPENSTACK SERVICE

Nova is the main Openstack services as it interact with all the services to set IAAS stack. Nova interact with Glance service to provided images for VM provisioning.It also interact with the Queue service via a nova-scheduler and nova-conductor API.It interacts with Keystone for authentication and authorization services. It also interacts with Horizon for web interface.

7. NOVA DEPENDENCE ON AMQP

AMQP is the messaging technology chosen by the OpenStack cloud.The AMQP sits between any two Nova components and allow them to communicate in a loosely coupled fashion. Nova Components use RPC to communicate to one another.It is build on the pub/sub paradigm to have the benefits. Decoupling between client and server(such that the client does not need to know where the servant's reference is) is a major advantage of AMQP[7].

8. ORCHESTRATION TASK IN NOVA

Nova-Conductor service plays an important role to manage the execution of workflows which involve the scheduler. Rebuild, resize, and building the instance are managed here. This was done in order to have better separation of responsibilities between what compute nodes should handle and what schedular should handle and to clean up the path of execution. In order to query the schedular in a synchronous manner it needed to happen after the API had returned a response otherwise API response times would increase and thats why conductor was chosen. And changing the schedular call from asynchronous to synchronous helped to clean up the code[8].

The earlier logic was complicated and the scheduling logic was distributed across the code.The earlier process was changed to the new sequence. Firstly, API receives request to build an instance. The, the API sends an RPC cast to conductor to build an instance.Next,the conductor sends an RPC call to schedular to pick a compute and waits for the response. If there is a Scheduler fail, it stops the build at the conductor. Lastly, the conductor sends an RPC cast to the compute to build the instance. If the build succeeds, stop here. If it fails then the compute sends an RPC cast to conductor to build an instance. This is the same RPC message that was sent by the API.

9. OPENSTACK NOVA SUPPORT

Earlier Openstack was supported on KVM but its support has been extended QEMU. Microsoft Hyper -V and Vmware ESXi too provide extended support. Nova has support for XenServer and XCP through XenAPI virtual layer.It also support bare metal deployment and provisioning from 'Ironic' version. This means it is possible to deploy virtual machines. By default, it will use PXE and IPMI to provision and turn on/off the machine. But from the Ironic version it support vendor-specific plugins which may implement additional functionality.

10. OPENSTACK NOVA IN BIGDATA

A use case has been developed to leverage OpenStack to perform big data analytics. OpenStack used cassandra database for columnar data structure. PostgreSQL for relational data structure.This use case also allows us to configure Hadoop distributed file system for large unstructured data. OpenStack Sahara has come up with core cloud components of Big data[9].

11. OPENSTACK NOVA AND OTHER COMPETITORS

The OpenStack nova does the same task as it is being done by AWS EC2, Google CE and Microsoft Azure VM. With respect to Beach marking the main difference is the cloud administrator can upload their images in OpenStack where as in AWS and Google Cloud storage, one need to use the pre-defined list.The AWS is used mainly as public cloud where as the Openstack can be used a private cloud. But OpenStack has an advantage of customizing our own cloud configuration which is not there in any of the vendor specific clouds.

12. CONCLUSION

Here, we discussed about the main components of OpenStack compute layer-Nova and how it is interacting with different other Openstack services like Swift, Horizon etc. We also showed use case of running big data problem on Openstack. We also discussed the compute service offerings provided by

cloud vendors other than OpenStack.This helped us to understand the the overall picture of Nova and how it fits to offer a unique enterprise level IAAS cloud solution.

REFERENCES

[1] Wikipedia, "Openstack nova wikipedia," accessed: 03-23-2017. [Online]. Available: https://en.wikipedia.org/wiki/OpenStack

[2] OpenStack, "Openstack nova official," accessed: 03-23-2017. [Online]. Available: https://docs.openstack.org/developer/nova/architecture.html

[3] Wikipedia, "Openstack nova bigdata," accessed: 03-23-2017. [Online]. Available: https://en.wikipedia.org/wiki/OpenStack

[4] P. Website, "Openstack nova on pepple website," accessed: 03-23-2017. [Online]. Available: http://ken.pepple.info/openstack/2011/04/22/openstack-nova-architecture/

[5] O. Website, "Openstack nova schedular," accessed: 03-23-2017. [Online]. Available: https://docs.openstack.org/kilo/config-reference/content/section_compute-scheduler.html

[6] I. Developers, "Openstack nova ibm," accessed: 03-23-2017. [Online]. Available: https://www.ibm.com/developerworks/cloud/library/cl-openstack-pythonapis/

[7] O. Website, "Openstack nova amqp," accessed: 03-23-2017. [Online]. Available: https://docs.openstack.org/developer/nova/rpc.html

[8] O. Website, "Openstack nova orchestrator," accessed: 03-23-2017. [Online]. Available: https://docs.openstack.org/developer/nova/conductor.html

[9] O. Website, "Openstack nova bigdata," accessed: 03-23-2017. [Online]. Available: https://www.openstack.org/summit/san-diego-2012/openstack-summit-sessions/presentation/big-data-on-openstack-a-rackspace-use-case

Heroku

YATIN SHARMA[1,*,+]

[1] School of Informatics and Computing, Bloomington, IN 47408, U.S.A.
[*] Corresponding authors: yatins@indiana.edu
[+] HID - S17-IR-2034

Paper-002, April 30, 2017

Heroku is a web application hosting platform-as-a-service cloud that enables developers to build and deploy application. It supports multiple programming languages including Ruby, Java, Node .js and Python. It is a simple and modular platform that allows developers to focus less on infrastructure/deployment and focus more on coding.

Keywords: Heroku, PaaS, Dyno, Dyno Manifold, Logplex, Toolbelt client, Procfile.

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IR-2034/report.pdf

1. INTRODUCTION

Heroku[1] is Platform as a Service(Paas)[2] that enables us to build and deploy applications in various programming language including Java, Python, Ruby, etc. It allows us to deploy web applications seamlessly as well as monitor and share with other developers instantly. It is enables rapid application development for the cloud, using the underlying platform infrastructure and software add-ons to build, deploy and monitor large and scalable web applications. It is built on top of Amazon Web Services and is owned by Salesforce.com [3].

2. BENEFITS

Heroku is language agnostics and provides great flexibility in choosing an appropriate programming language to develop the web application. It has core support for Clojure, Ruby, Java, Python, Node.js, Scala and Play, Go, and PHP. Apart from these, any other language can also be supported by using a feature called buildpacks. Heroku provides a lot of flexibility in managing the applications after deployment using the Heroku command line tool running on the client machine or on the Heroku Infrastructure. Heroku uses Git[4] as the means for deploying applications to its servers. Being owned by Salesforce.com[3], it has a feature called Heroku Connect enabling interaction between the applications and Salesforce API.

3. ARCHITECTURE

The Heroku architecture consists of platform stack containing various runtime libraries, OS, and underlying infrastructure. A Heroku application can be thought of a multiple processes, each consuming resources like a normal UNIX process, that run on the Heroku Dyno manifold. Heroku defines each process through configuration file called Procfile- which is a text file, placed in root of the application and contains the format describing how the application will run. Fig. 1. below describes the high level architecture of the Heroku platform.

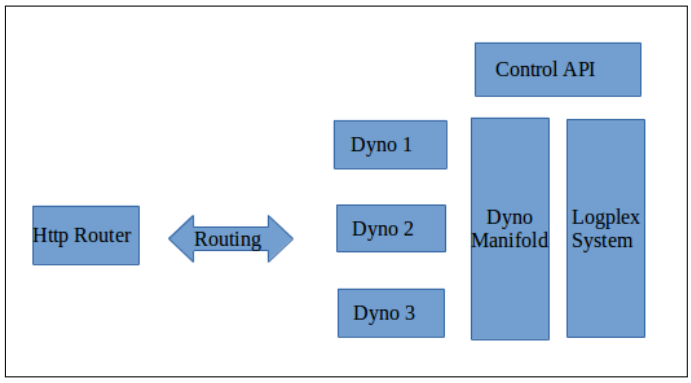


Fig. 1. High Level Architecture of Heroku platform

3.1. Process Management

The unit of work in Heroku framework is called Dyno. It can be thought of as packaged running version of the code that the application interacts with. Dynos are responsible for receiving web requests, writing an output and connecting to application resources such as databases. They are fully isolated containers running on the Dyno Manifold, which is the building block for execution environment. Dyno Manifold is responsible for process management, isolation,scaling, routing, distribution that are necessary for to run the web and worker processes. It is also fault tolerant and distributed in nature. If any dynos fails, the manifold restarts them automatically, hence removing a lot of maintenance hassles.Dynos are capable of serving many

request per second and execute in complete isolation from each other. Each dyno gets its own virtual environment that it can use to handle its own client requests. Dynos use LXC to provide container like behavior to achieve complete isolation from one another.There is a memory restriction of 1.5 GB per dyno, beyond which the dyno is rebooted with a Heroku error which could lead to a memory leak.

3.2. Execution Flow

Process type is the declaration of the command that defines the structure to be used while instantiating a process. Heroku has two process types- web process, which is responsible for handling HTTP client requests and the worker process, which is responsible for executing other tasks such as customer jobs of running background jobs and queuing tasks.

3.3. Logging Architecture

Heroku logplex system provides a flexibility facility by giving us an overall view of the application runtime behavior. It forms the basis of the Heroku logging infrastructure. It routes log streams form various sources into single output(for example archival system). The logplex system keeps the most recent data (typically 1500 logs) that are important to extract relevant information from the application being run.

3.4. Http Routing

Routing Mesh are responsible for routing the web requests to the appropriate web process dyno. It is also responsible for seeking the application's web dynos within the dyno manifold and forwarding the HTTP web requests to one of them. The routing mesh uses a round robin algorithm to distribute the request acroos various dynos. Since the dynos could be running in distributed manner, the routing mesh manages the access and routing internally and none of the dynos are statically addressable. Heroku also supports multithreaded and asynchronous application, accepting many network connections to process client requests.

3.5. Heroku Interfaces

Heroku provides the developer with the flexibility to control various aspects of the application through various control surfaces such as process management, routing, logging, scaling, configuration, and deployment. These are available as a command-line interface (CLI), a web-based console, and full REST API.

4. ADDONS

Heroku supports vast amount of third party addons[5] that any Heroku user can instantly provision as an attachable resource for their application. For example, if the application needs a PostgreSQL[6] database, it can be done in Heroku. Addons work through Heroku's environment variables. Each time an addon is added to the application, the application will automatically be assigned new environment variables which specify any configurations required to interact with the new addon. Just like Dynos, Addons are easy to add, up-grade, downgrade, and remove without requiring any downtime for the applications.

5. HEROKU PLATFORM API

Heroku platform API is a tool that allows to call Heroku platform services, create applications, and plug in new add-ons by simply using HTTP. It gives the developers complete control over their application. The three components that define the behavior of the API are : 1)Security 2)Schema 3)Data. The client accesses the API using standard methods defined for HTTP. The API then acts on the request and returns the result in JSON format.

6. SECURITY

Heroku employs various measures to ensure that the application and data stores within the platform are secure from external attacks,thefts and hacks. Heroku enforces SSH[7] protocol to encrypt the source code while they are getting pushed into the Heroku environment. Any application that runs on Heroku is in complete isolation from another, so that no two application can see each other getting executed. It also restricts applications from making local network connection between hosts. It enables data security by keeping the data in access controlled databases.

7. GETTING STARTED

There are few prerequisites that we need to perform before we can start using Heroku: 1) Get Heroku account 2) Install Heroku toolbelt client[8] 3) Set up SSH for the user account. Heroku toolbelt is the client software required to work with the Heroku platform and contains the following component: Heroku Client, Foreman and Git. Step-by-step procedures to download, install and get started with Heroku can be found on-line.[9]

8. CONCLUSION

Heroku offers a Platform as a Service (PaaS) in which the servers and filesystems are completely abstracted away. It gives provides an environment where pushing the code and basic configuration can get an application running. It provides a complete developer experience and an application runtime. It manages all that in scalable and highly maintainable fashion. It's logging service as well as instant scaling power allows scaling up to support demand from users, and scaling down when high straffic has stopped.

ACKNOWLEDGEMENTS

The author thanks Prof. Gregor von Laszewski for his technical guidance.

REFERENCES

[1] "Cloud application platform | heroku," Web Page, online; accessed 13-Mar-2017. [Online]. Available: https://www.heroku.com/

[2] "Platform as a service - wikipedia," Web Page, online; accessed 13-Mar-2017. [Online]. Available: https://en.wikipedia.org/wiki/Platform_as_a_service

[3] "Salesforce.com | the customer sucsess platform to grow your business," Web Page, online; accessed 3-April-2017. [Online]. Available: https://www.salesforce.com/

[4] "Github," Web Page, online; accessed 13-Mar-2017. [Online]. Available: https://github.com/

[5] "Add-ons - heroku elements," Web Page, online; accessed 3-April-2017. [Online]. Available: https://elements.heroku.com/addons

[6] "Heroku postgres - add-ons - heroku elements," Web Page, online; accessed 3-April-2017. [Online]. Available: https://elements.heroku.com/addons/heroku-postgresql

[7] "Secure shell- wikipedia," Web Page, online; accessed 17-Mar-2017. [Online]. Available: https://en.wikipedia.org/wiki/Secure_Shell

[8] "Heroku cli | heroku dev center," Web Page, online; accessed 17-Mar-2017. [Online]. Available: https://devcenter.heroku.com/articles/heroku-cli

[9] “Getting started on heroku | heroku dev center,” Web Page, online; accessed 3-April-2017. [Online]. Available: https://devcenter.heroku.com/start

D3

PIYUSH SHINDE[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: piyushsshinde1992@gmail.com*

Paper-2, April 30, 2017

Data Driven Documents (D3) is a open source JavaScript library used to create dynamic, interactive visualizations on a webpage. D3 uses HTML, SVG and CSS to create visualizations with a data-driven approach to Document Object Model (DOM) manipulation, enabling users to utilize the full capabilities of modern browsers and the freedom to design the right visual interface for their data [1]. This paper provides a brief introduction to D3 and its various features. It also discusses D3's use cases, advantages and limitations.

Keywords: Document Object Model, Data Visualization, Chart

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2035/report.pdf

1. INTRODUCTION

Several graphical forms can be used to envision large quantitative data sets such as graphs, charts, maps and diagrams. These data sets are increasing exponentially since the advent of the World Wide Web, making the task of efficient data visualization more challenging. As a result, web browsers are considered as ideal platforms for visualizing large data sets.

Designers often employ multiple tools simultaneously for building visualizations like HTML for page content, CSS for aesthetics, JavaScript for interaction, SVG (Scalable Vector Graphics) for vector graphics [2]. One of the most important advantages is the uninterrupted cooperation between most of the technologies using the web as a platform, which is enabled by a document object model (DOM). The DOM reveals the ordered structure of a web page, aiding it's manipulations.

Data Driven Documents (D3) is an open source JavaScript library, which helps in efficient manipulation of the document object models based on data. It was released in 2011 by Heer, Ogievetsky and Bostock, then part of Computer Science Department of Stanford University. D3 provides a toolkit for visualizing data using web standards such as HTML, SVG, and CSS.[1].

D3 resembles to document transformers such as jQuery [3] that simplify the act of document transformation in web browsers. D3 uses the DOM's standard SVG syntax, that shows similarities with the graphical abstractions used in graphics libraries such as Processing and Raphaël [4]. D3 is a generalization of Protovis [5], and through helper modules more complex visualizations can be achieved efficiently.

D3's main features include selections, transitions and update, enter and exit functions. We will glance through these features in the next section.

2. FEATURES

D3's basic operand is the selection. Operators act on selections, modifying content. Data joins bind input data to elements, producing enter and exit sub-selections for the creation and destruction of elements with respect to data. Animated transitions interpolate attributes and styles smoothly over time [2].

2.1. Selections

Selections allow the user to select and manipulate HTML elements in a very simple way. D3 adopts the W3C Selectors API [6] that contain predicates to select elements by tag, class, unique identifier, attribute, containment, or adjacency [7]. Unique selection methods like union and intersection can be used on these predicates. Multiple operations can be performed after selecting an element by chaining the operations.

D3 provides select and selectAll methods for single and multiple element selections. The former selects only the first element that matches the predicates, while the latter selects all matching elements in document traversal order [2].

2.2. Data

Styles, attributes, and other properties are represented as functions of data in D3. They are simple, but powerful. D3 provides many built-in reusable functions and function factories, such as graphical primitives for area, line, and pie charts.

The data operator binds input data to selected nodes. Data is specified as an array of values such as numbers, strings or objects, and each value is passed as the first argument, along with the numeric index to selection functions. By default, data is joined to elements by index, the first element in the data array is passed to the first node in the selection, the second element to the second node, and so on. Once the data has been bound to

the document, you can omit the data operator, D3 retrieves the previously-bound data. This allows you to recompute properties without rebinding [7].

2.3. Enter, Update and Exit

D3's Enter and Exit selections, aid users to create new nodes for incoming data and remove outgoing nodes that are no longer required.

When data is bound to a selection, each element in the data array is paired with the corresponding node in the selection. In case of fewer nodes as compared to elements of the data array, the extra data elements form the enter selection, that can be instantiated by appending to the enter selection as shown in figure 1.

Updating nodes are the default selection—the result of the data operator [7]. In case of skipped enter and exit selections, D3 automatically selects only the elements for which corresponding data exists. Properties that are constant for the life of the element are set once on enter, while dynamic properties are recomputed per update [2]. The initial selection can be divided into three parts: the updating nodes to modify, the entering nodes to add, and the exiting nodes to remove. Handling these three cases separately, precisely define the operations that run on each node, thereby optimizing performance and offering greater control over transitions.

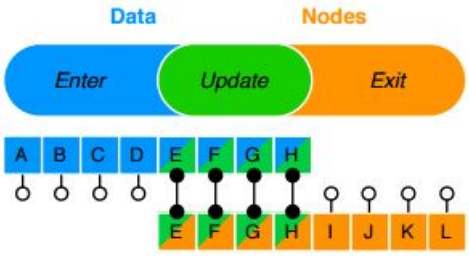


Fig. 1. New data (blue) joining old nodes (orange) results in three sub-selections: enter, update and exit [7].

2.4. Transitions

D3's focus on transformation extends naturally to animated transitions. Transitions gradually interpolate styles and attributes over time. D3's interpolators support both primitives, such as numbers and numbers embedded within strings (font sizes, path data, etc.), and compound values. D3's interpolator registry can even be registered to support complex data structures.

D3 reduces overhead by adjusting only the attributes that change and allows greater graphical complexity at high frame rates. D3 allows sequencing of complex transitions via events. D3 does not replace the browser's toolbox, but exposes it in a way that is easier to use [7].

3. USE CASES

The examples tab of the official D3 website displays more than 400 examples of data visualizations, built using D3 [8]. Examples include visualizations for newspapers, games, libraries and tools suggesting D3's reliability and ability to transform even the most complex data into clear visualizations. These visualizations run interactively inside web browsers and are available for anyone who wants to visualize data.

Many of them are accompanied by a full documentation of the steps to manipulate documents based on data. The most commonly used types of data visualization are histogram, area chart, line chart, multi series line chart, bar chart, scatter-plot, pie chart, graphs, trees and maps [8]. D3 also shows visualizations updating in real time.

Here are three real world examples of simplified data visualizations using D3.

3.1. Wimbledon 2013 Data Visualisations

This example displays a series of ten different data visualizations of the Wimbledon tennis tournament created by Peter Cook [9]. The original data was collected from British tennis data website [10].

Amongst the 10 visualizations, 3 of them include a circular match tree, a bubble chart and a horizontal histogram. The circular match tree displays concentric circles labelled as the different rounds of the Wimbledon 2013 as shown in figure 2. Each concentric circle displays the player names. The result was of each match in each round is displayed by hovering the mouse over the names. The winner's name is displayed in the center of the circle.

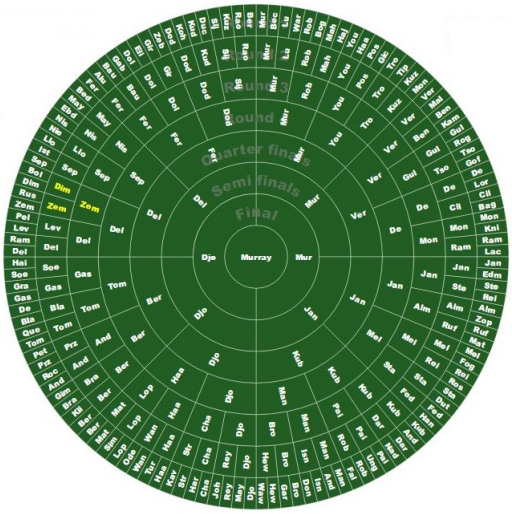


Fig. 2. Wimbledon 2013: Circular Match Tree [11].

The bubble chart displays pair of bubbles connected by an arrow as shown in figure 3. Each arrow indicates a match where the winner had a lower ATP ranking than the loser. The size of each bubble is proportional to the ATP points of the player. Hovering the mouse over a bubble displays both the players and their ATP points with the result of the match.

The histogram displays the list of the top 32 players of Wimbledon 2013 as shown in figure 4. It displays the number of matches won, sets won, games won and ATP ranking of all the 32 players by clicked on the respective tabs [13].

3.2. Earthquakes in Chile since 1900

This example helps us visualize the most important seismic events in Chile since 1900 [14]. The earthquake data was retrieved from the ANSS Composite Catalog and joined with the Centennial Catalog from the USGS.

A seismic event is displayed as a circle with it's occurrence year as shown in figure 5. The radius and the color of the circles are a function of the earthquake magnitude.

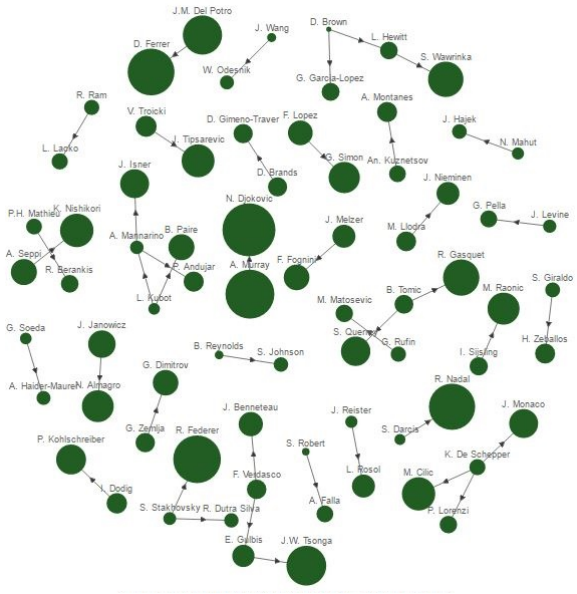


Fig. 3. Wimbledon 2013: Bubble Chart [12].

image missing

Fig. 4. Wimbledon 2013: Top 32 Players [13].

3.3. Visualizing the Racial Divide

This example displays a visualization that highlights the impact of the segregation that exists in many US cities today based on race using d3 and force-directed maps [15]. The cities include Milwaukee, Chicago, St. Louis, Syracuse, Dayton, Pittsburgh, Denver, Columbus, Kansas City, Oklahoma City, Wichita, Memphis, Baltimore and Charleston. Each city is made up of tracts from the 2010 Census.

The Census tracts are pushed away from neighboring tracts based on the change in proportion of white and black populations between each neighboring tract as shown in figure 6.

Tracts having similar racial mix as their neighbors form groups. Spaces occur where there is a significant change in the racial makeup between neighboring tracts. The space is proportional to the change in racial composition between neighbors.

4. BENEFITS AND LIMITATIONS

D3 is an open source project, with it's source code readily available on Mike Bostock's github account [17]. It is free to use, which sets it apart from similar data visualization JavaScript libraries such as amCharts and FusionCharts, which need paid licenses. D3's development is supported by a big online community, where users can contribute to it's examples making it an exhaustive resource of various data visualizations.

D3's is also a drawing library unlike conventional data visualization tool-kits, that simply create data visualizations. D3 supports dynamic visualizations, as opposed to mere static visualizations.

D3 can be started using just one line of code, which is not the case with other data visualization tool-kits, often requiring a long installation procedure and periodic updating. D3's compatibility with web standards provide an added advantage that visualizations can be shared and viewed without the need for additional plug-ins. It is based on JavaScript which is compatible with browser's built-in debugger. This facilitates it's fast and easy debugging.

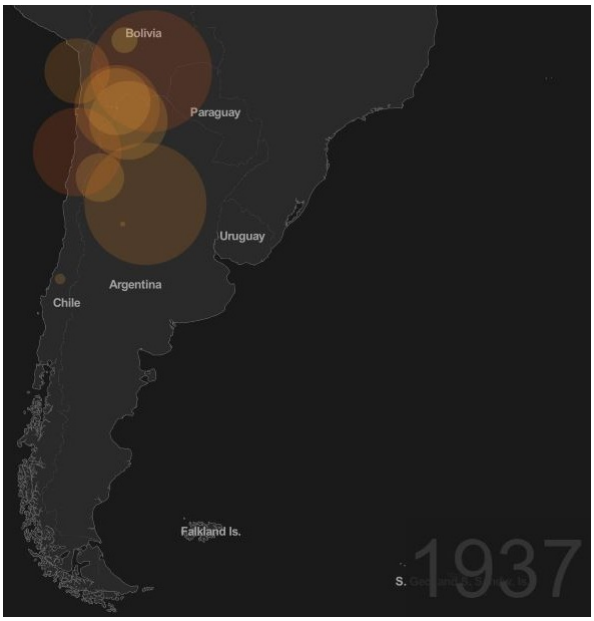


Fig. 5. Earthquakes in Chile since 1900: Map [14].

Fig. 6. Visualizing The Racial Divide: Chicago [16].

Amongst several advantages D3 has a few limitations. It's learning curve can be fairly steep. This is partly explained by the need for in depth knowledge of web standards, especially of SVG. Furthermore, the helper modules required for data transformations need studying too.

Unlike JavaScript libraries such as Google Charts, D3 does not offer in-built (standard) charts. This makes it less efficient as compared to other tool-kits with custom graphs.

5. USEFUL RESOURCES

The complete documentation of D3.js, including instructions to download and install it's latest release, examples, and several tutorials, in a systematic method are available on the main website of D3.js [7]. It is also a useful resource to D3's core concepts, techniques, blogs, books, courses, talks, videos and meetups [18].

The paper titled "D^3: Data-Driven Documents", compares D3 to existing web-based methods for visualization. It demonstrates how D3 is at least twice as fast as Protovis, through performance benchmarks. It also describes D3's potential for dynamic visualization [2].

Another website provides a complete path to create interactive visualization using D3.js [19]. It provides few real world examples and steps to create basic pie charts, animated bar charts and map.

6. CONCLUSION

The increasing popularity of JavaScript, has caused a major shift in the direction of web development with reduced dependencies on plug-ins. Consequently, developers are trying rely on the web browsers alone to avoid dependence on external plug-ins. This reduces the possibility of bugs or incompatibility issues as well the need to update. As an added benefit, developers can work with the existing web standards, increasing D3's compatibility with technologies like HTML, CSS and JavaScript. This makes preparation of data visualizations easy and readily available to everyone. D3.js is an toolkit for efficiently creating complex visualizations based on large datasets. It can be used for solving real world problems by creating visualisations to better infer trends in complex large data sets.

7. ACKNOWLEDGMENTS

This project was a part of the Big Data and Software and Projects (INFO-I524) course. I would like to thank Professor Gregor von Laszewski and the associate instructors for their help and support during the course.

REFERENCES

[1] Mike Bostock, "Home," Web Page, accessed: 2017-03-24. [Online]. Available: https://github.com/d3/d3/wiki

[2] M. Bostock, V. Ogievetsky, and J. Heer, "D3: Data-driven documents," *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2011. [Online]. Available: http://vis.stanford.edu/papers/d3

[3] jQuery Foundation, "jQuery," Web Page, accessed: 2017-03-24. [Online]. Available: https://jquery.com/

[4] Dmitry Baranovskiy, "Raphaël," Web Page, accessed: 2017-03-24. [Online]. Available: http://dmitrybaranovskiy.github.io/raphael/

[5] Mike Bostock, "Protovis," Web Page, accessed: 2017-03-24. [Online]. Available: http://mbostock.github.io/protovis/

[6] Anne van Kesteren and Lachlan Hunt, "W3C," Web Page, accessed: 2017-03-24. [Online]. Available: https://www.w3.org/TR/selectors-api/

[7] Mike Bostock, "D3 Data Driven Documents," Web Page, accessed: 2017-03-24. [Online]. Available: https://d3js.org/

[8] Mike Bostock, "Gallery," Web Page, accessed: 2017-03-24. [Online]. Available: https://github.com/d3/d3/wiki/Gallery

[9] Peter Cook, "Wimbledon 2013 Data Visualisations," Web Page, accessed: 2017-03-24. [Online]. Available: http://charts.animateddata.co.uk/tennis/index.html

[10] Joseph, "Tennis-Data.co.uk," Web Page, accessed: 2017-03-24. [Online]. Available: http://www.tennis-data.co.uk/wimbledon.php

[11] Peter Cook, "Wimbledon 2013 Circular Match Tree," Web Page, accessed: 2017-03-24. [Online]. Available: http://charts.animateddata.co.uk/tennis/matchTree.html

[12] Peter Cook, "Wimbledon 2013 David and Goliath," Web Page, accessed: 2017-03-24. [Online]. Available: http://charts.animateddata.co.uk/tennis/davidgoliath.html

[13] Peter Cook, "Wimbledon 2013 Top 32 Players," Web Page, accessed: 2017-03-24. [Online]. Available: http://charts.animateddata.co.uk/tennis/top32.html

[14] Pablo Navarro, "Earthquakes in Chile since 1900," Web Page, accessed: 2017-03-24. [Online]. Available: http://pnavarrc.github.io/earthquake/

[15] Jim Vallandingham, "Visualizing The Racial Divide," Web Page, accessed: 2017-03-24. [Online]. Available: http://vallandingham.me/racial_divide/

[16] Jim Vallandingham, "Visualizing The Racial Divide," Web Page, accessed: 2017-03-24. [Online]. Available: http://vallandingham.me/racial_divide/#ch

[17] Mike Bostock, "Mike Bostock," Web Page, accessed: 2017-03-24. [Online]. Available: https://github.com/mbostock

[18] Mike Bostock, "Tutorials," Web Page, accessed: 2017-03-25. [Online]. Available: https://github.com/d3/d3/wiki/Tutorials

[19] Analytics Vidhya, "Newbie to D3.js Expert: Complete path to create interactive visualization using D3.js," Web Page, accessed: 2017-03-24. [Online]. Available: https://www.analyticsvidhya.com/learning-paths-data-science-business-analytics-business-intelligence-big-data/newbie-d3-js-expert-complete-path-create-interactive-visualization-d3-js/

An overview of the open source log management tool - Graylog

RAHUL SINGH[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: rahpsing@iu.edu*

April 30, 2017

Graylog is an open source log management tool that allows an organization to collect, organize and analyze large amounts of data from its network activity. It enhances the basic log management functionality by providing network traffic analysis, lucene syntax based search, drill-down analysis of data using field statistics and generates trigger actions based alert notifications. It integrates with other open source technologies to address a larger distributed system.

Keywords: Cloud, I524

https://github.com/cloudmesh/sp17-i524/blob/master/paper2/S17-IR-2036/report.pdf

1. INTRODUCTION

Graylog [1] allows management of an organization's computing resources in a consistent way. It allows us to centrally collect and manage log messages of an organization's complete infrastructure. A user can perform search on terabytes of log data to discover number of failed logins, find application errors across all servers or monitor the activity of a suspicious user id. Graylog works on top of *ElasticSearch* [2] and *MongoDB* [3] to facilitate this high availability searching. It provides a *Lucene* [4] like query language, a processing pipeline for data transformation, alerting abilities and much more. Graylog enables organizations, at a fraction of the cost, to improve IT operations efficiency, security, and reduce the cost of IT [5].

2. ARCHITECTURE

Graylog is written in Java and uses a few key open source technologies like ElasticSearch and MongoDB. Additionally, for a larger setup *Apache Kafka* [6] or *RabbitMQ* [7] could be integrated to implement queueing. A basic Graylog cluster consists of the following components:

Graylog server - It is the actual log processor system also responsible for implementing security. The Graylog server nodes shall be operated on the fastest CPU's available.

Graylog web UI - It is the Graylog web user interface where one can view histograms, dashboards and create alerts.

MongoDB - MongoDB stores Graylog configuration information and the non queried log messages. It's prime purpose in the architecture is Metadata Management [8].

ElasticSearch - ElasticSearch is useful for storing actual log data and perform search operations on them. ElasticSearch nodes should have as much RAM as possible and the fastest disks linked to them. Messages are only stored on ElasticSearch nodes. If we have data loss on ElasticSearch, all messages are gone – except if the administrator has created backups of the indices.

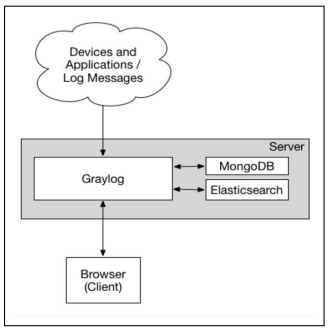


Fig. 1. Graylog minimum architecture [9]

3. GRAYLOG USE CASES

3.1. Computer And Network Security

The best way for intrusion detection is to monitor activity of all the devices in the network. Graylog allows a user to keep track of all failed logins, rejected network connections or exceptions

in the flow of the application. It also allows a user to integrate other *Intrusion Detection Systems (IDS)* [10] to correlate detected activity with logs from all across the infrastructure [1].

3.2. Centralized IT management

Logging into every system and parsing the plain text log files to find meaningful data is an arduous task for any IT engineer. Graylog allows us to centrally collect all syslog and eventlog messages of the complete infrastructure thus allowing to solve production issues in real time. It does so by allowing the administrator to setup alerts for any trigger actions like performance degradation or exceptions.

3.3. Development and DevOps

Graylog allows on demand monitoring of distributed applications by giving tiered access to any developer in the organization to view system and application logs. It works on top of elastic search nodes to facilitate search operation on terabytes of log data in a matter of milliseconds. In case of a customer operation resulting in an error, a developer shall need to search the logs with the customer id and locate the relevant logs to find the root cause of the problem.

4. GRAYLOG MODULES

4.1. Sending in log data

Graylog needs a log source to serve its purpose. The message input section is launched from the System - Inputs section from the web interface (or the *REST API* [11]) and can be configured without the need to restart any part of the system [12].

Graylog is able to accept and parse *RFC 5424* [13] and *RFC 3164* [14] compliant syslog messages and the Graylog Extended Log Format (GELF) [12]. For any devices on the network that do not publish RFC compliant syslog messages we need to make use of the plaintext messages. The GELF is a log format that avoids the shortcomings of classic plain syslog by providing optional compression, fixed structure and limits payload length to 1024 bytes. Syslog is preferred for direct logging by machines in the network while GELF is suitable for logging from within the applications.

4.2. Search

As Graylog uses ElasticSearch to facilitate searching, the search syntax followed is very close to the Lucene syntax which is the underlying implementation of Elastic Search. By default, search is performed over all fields unless a specific field is specified in the search query. Search features like fuzzy, wildcard, range searches provided by Elastic Search API can be used to gain deeper insight to data. Graylog also provides a time frame selector which defines the time range over which the search shall be performed. The time selector could be relative, absolute or keyword based. Graylog also allows a user to save his searches to view it later. A user needs to save his search by a unique name and could load it later from the system, under the saved search selector.

Graylog automatically constructs a histogram for the search results. The histogram depicts the concise number of messages received grouped by a certain time period that is adjustable. Based on a user's recent search query, graylog also allows you to distinguish data that are not searched upon very often and thus can be archived on cost effective storage drives.

4.3. Log Streams

Graylog allows a user to create a set of rules to route messages into user defined categories. A user could create a stream called 'Database errors' and create a rule to direct all messages with the source attribute as 'database' to that stream. Thus, the stream 'Database errors' shall catch every error message from the system's database hosts. A message shall be routed into every stream that has the corresponding matching rule for the message. A message thus, can be part of many streams and not just one [15].

4.4. Alerts

Graylog alerts are periodical searches that can trigger some notifications when a defined condition is satisfied [16]. To get notified when more than 50 exceptions occur in the range of a minute, an alert can be created with the desired conditions. While defining alerts, a user can also specify the method of notification once the alert condition is met. Notifications can be obtained by an email or by an HTTP request to an endpoint in the system.

4.5. Dashboards

Graylog provides visualization through creation of dashboards that allows a user to build pre-defined views on his data to assemble all of his important data only a single click away [17]. Any search result or metric shall be added as a widget on the dashboard to observe trends in one single location. A user can also add search result metrics like result count, statistical values, field value charts and stack charts to the dashboard. These dashboards can also be shared with other users in the organization.

4.6. Graylog REST API

Both configuration settings and log data are available through the Graylog REST API. The Graylog web interface uses Graylog Rest API internally to interact with the Graylog cluster. Graylog REST API could be used for automation or integrating Graylog into another system, such as monitoring or ticket systems [18]. Thus a network administrator can easily integrate Graylog into his evolving architecture and build reports and analysis.

4.7. Filtering messages

Graylog can use *Drools* [19] to evaluate all incoming messages against a user defined rules file. To discard any message before its written to elastic search or to forward it to another system, one can use Drools rules to perform custom filtering [20].

5. COMPARISION WITH SPLUNK

Splunk [21] is considered as one of the best log management tools available and is Graylog's biggest competitor. Splunk started as a log analysis tool but has now grown into a full machine generated data processing platform. Splunk works on almost every format of log data unlike graylog, but has a higher setup cost. To deploy Splunk in a high scale environment, a user needs to install and configure a dedicated cluster. Table below represents a comparision of Graylog and Splunk for a few basic factors.

6. CONCLUSION

Graylog makes big data analytics affordable by providing an open source solution that allows organizations to realize the benefits of collecting and analyzing log data and thus improving

Table 1. Comparision of Graylog and Splunk [22] [23]

Parameter	Graylog	Splunk
Business Model	Opensource	Commercial software
Setup Time	Needs time	No time
Learning Curve	Difficult	Simple
Filetypes	syslog,gelf	Many
Security	Good	Good
Apps Supported	Low	Very high

operational efficiency at a reduced cost of IT. It provides an effective set of features to be adapted by any small to medium size organization. Alert notifications, sharing of dashboards and message filtering provide most features that any network administrator desires from a log management system. Being an open source tool it is cost effective compared to other log systems. Graylog provides centralized monitoring and management of large scale distributed systems from a single point of control. However, it needs an environment to be setup before it can be operational. It has a steep learning curve with the responsibility of managing the MongoDB and ElasticSearch instances being completely managed by the user. Hence, the choice to choose Graylog as an organization's log management tool directly relies on the resources in terms of either time or money that it chooses to employ.

REFERENCES

[1] "Graylog | open source log management," webpage, accessed : 03-19-2017. [Online]. Available: https://www.graylog.org/

[2] Wikipedia, "Elasticsearch - wikipedia," webpage, accessed : 04-09-2017. [Online]. Available: https://en.wikipedia.org/wiki/Elasticsearch

[3] Wikipedia, "Mongodb - wikipedia," webpage, accessed : 04-09-2017. [Online]. Available: https://en.wikipedia.org/wiki/MongoDB

[4] "Apache Lucene - welcome to apache lucene," webpage, accessed : 04-09-2017. [Online]. Available: https://lucene.apache.org/

[5] "High tech grunderfonds | graylog raises $2.5 million to expand opensource big data analytics platform," webpage, Feb 2015, accessed : 04-09-2017. [Online]. Available: http://high-tech-gruenderfonds.de/en/graylog-raises-2-5-million-to-expand-open-source-big-data-analytics-platform/

[6] "Apache Kafka," webpage, accessed : 04-09-2017. [Online]. Available: https://kafka.apache.org/

[7] "Rabbit MQ - messaging that just works," webpage, accessed : 04-09-2017. [Online]. Available: https://www.rabbitmq.com/

[8] Severalnines, "High availibility log processing with graylog,mongodb and elastic search," webpage, Mar 2016, accessed : 03-19-2017. [Online]. Available: https://severalnines.com/blog/high-availability-log-processing-graylog-mongodb-and-elasticsearch

[9] "Architectural consideration - graylog 2.2.1 documentation," webpage, accessed : 03-19-2017. [Online]. Available: http://docs.graylog.org/en/2.2/pages/architecture.html

[10] Wikipedia, "Intrusion detection system - wikipedia," webpage, accessed : 04-09-2017. [Online]. Available: https://en.wikipedia.org/wiki/Intrusion_detection_system

[11] Wikipedia, "Representational state transfer-wikipedia," webpage, accessed : 04-09-2017. [Online]. Available: https://en.wikipedia.org/wiki/Representational_state_transfer

[12] "Sending in log data - graylog 2.2.1 documentation," webpage, accessed : 03-20-2017. [Online]. Available: http://docs.graylog.org/en/2.2/pages/sending_data.html

[13] R. Gerhards, "https://www.ietf.org/rfc/rfc5424.txt," webpage, Mar 2009, accessed : 04-09-2017. [Online]. Available: https://www.ietf.org/rfc/rfc5424.txt

[14] C. Lonvick, "https://www.ietf.org/rfc/rfc3164.txt," webpage, Aug 2001, accessed : 04-09-2017. [Online]. Available: https://www.ietf.org/rfc/rfc3164.txt

[15] "Streams - graylog 2.2.1 documentation," webpage, accessed : 03-20-2017. [Online]. Available: http://docs.graylog.org/en/2.2/pages/streams.html

[16] "Alerts - graylog 2.2.1 documentation," webpage, accessed : 03-20-2017. [Online]. Available: http://docs.graylog.org/en/2.2/pages/streams/alerts.html

[17] "Dashboards - graylog 2.2.1 documentation," webpage, accessed : 03-20-2017. [Online]. Available: http://docs.graylog.org/en/2.2/pages/dashboards.html

[18] "Graylog rest api - graylog 2.2.1 documentation," webpage, accessed : 03-21-2017. [Online]. Available: http://docs.graylog.org/en/2.2/pages/configuration/rest_api.html

[19] "Drools - drools -business rules management system(java, open source)," webpage, accessed : 04-09-2017. [Online]. Available: https://www.drools.org/

[20] "Blacklisting - graylog 2.2.1 documentation," webpage, accessed : 03-20-2017. [Online]. Available: http://docs.graylog.org/en/2.2/pages/blacklisting.html

[21] "Operational intelligence, log management, application management, enterprise security and compliance | splunk," webpage, accessed : 04-09-2017. [Online]. Available: https://www.splunk.com/

[22] T. Weiss, "The 7 log management tools you need to know | takipi blog," webpage, Apr 2014, accessed : 04-09-2017. [Online]. Available: http://blog.takipi.com/the-7-log-management-tools-you-need-to-know/

[23] S. Yegulalp, "Open source graylog puts splunk on notice | infoworld," webpage, Feb 2015, accessed : 04-09-2017. [Online]. Available: http://www.infoworld.com/article/2885752/log-analysis/open-source-graylog-puts-splunk-on-notice.html

AUTHOR BIOGRAPHIES

Rahul Singh received his B.E. (Computer Engineering) from University of Mumbai, India. He is currently pursuing his Masters in Computer Science at Indiana University Bloomington.

Jupyter Notebook vs Apache Zeppelin - A comparative study

SRIRAM SITHARAMAN[1,*]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: srirsith@iu.edu*

April 30, 2017

With the development of new technologies for the purpose of exploring, analysing and visualizing big datasets, there exists a growing demand of a collaborative platform that combines the process of data analysis and visualization.The idea of computer notebooks has been around for a long time with the launch of Matlab and Mathematica. Two such platforms: Apache Zeppelin and Jupyter notebook are taken in to consideration for comparison. Both of them were ranked against characteristics such as their ability to support multiple programming techniques, multi-user support and integrated visualization.

Keywords: Jupyter, Zeppelin, Notebook, Comparison

https://github.com/cloudmesh/classes/blob/master/docs/source/format/report/report.pdf

CONTENTS

1. INTRODUCTION - JUPYTER NOTEBOOK

Jupyter notebook, part of project Jupyter is the third version of IPython notebook. It is a web based interactive development environment and supports multiple programming languages (Python, Julia, R etc.) [1]. It started supporting Julia, Python and R in its initial release, hence the term Jupyter. The web based interative environment provided by Jupyter is facilitated through a notebook interface which contains the programming code written by the user as well mark down elements and outputs that are generated as result of the written code[2]. Hence, Jupyter Notebook documents allows a seemless view of the code written and the corresponding output (graphs,tables,etc.). The Jupyter Notebook app is a client-server application as shown in the figure 1 that can be run on a local machine for personal use and can be accessed without the internet, or can be installed in a remote server which can be accessed via an internet connection. The final component of the Jupyter Notebook is the kernel which performs the execution of code written by user. Jupyter comes with a native Ipython kernel (supporting Python), and also supports 80+ programming languages' kernels [3].

2. INTRODUCTION - APACHE ZEPPELIN

Apache Zeppelin [5], similar to Jupyter notebook, is also a web based interactive notebook but aimed towards supporting big data and data analytics. It supports multiple programming languages. It is useful in exploration of data, creation of visualizations and sharing insights, as web pages, with various stakeholders involved in a project. It supports a range of programming languages like Scala, Spark SQL, Shell, etc with the default being scala. Zeppelin has an architecture similar to Jupyter notebook with an exception that notebook in Zeppelin supports the integration of multiple programming languages in a single notebook. Zeppelin's notebook is shipped with some basic charts which can be used to visualize output generated by any supported programming language. Apart from that, it has the option of pivot charts and Dynamic forms that can be generated inside the notebook interface.

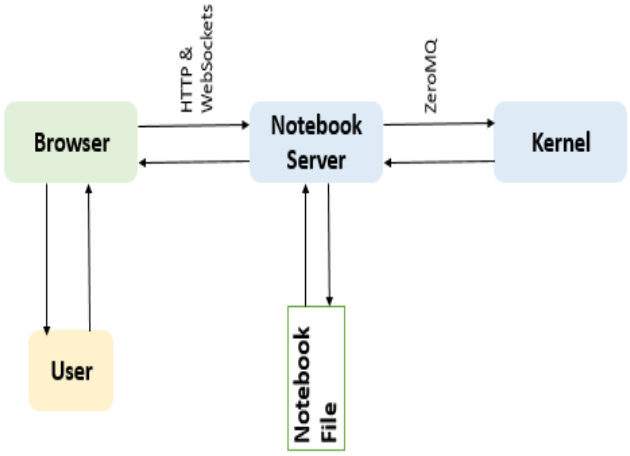


Fig. 1. Jupyter Architecture. The main components of the architecture comprises of the browser (user interface) which contacts with the Notebook server. The Notebook server can connect with multiple kernels.[4]

3. COMPARISON

This section would compare the following aspects of Apache Zeppelin and Jupyter notebook:

1. Interpreter configuration
2. Interface
3. Supported Languages
4. Visualization
5. Multi-user capability
6. Community Support

3.1. Interpreter configuration

Zeppelin interpreter is a language/data-processing-backend that can be plugged into Zeppelin notebook. Currently, Zeppelin supports over 30 interpreters such as Scala (with Apache Spark), Python (with Apache Spark), Spark SQL, JDBC, Markdown, Shell, etc [6]. Zeppelin has a separate Interpreter configuration page that you can be accessed for multiple language parameters. For instance, Spark home directory as well as spark master string can be modified to our preference. It would create the Spark Context automatically so you don't need to deal with it in each notebook.For example, to use Scala code in Zeppelin, "%spark" has to be included to load the interpreter. Apache Zeppelin provides several interpreters as community managed interpreters [7] which can be installed at once if **netinst** binary package has been installed. It also supports installation of 3rd party interpreters.

For Jupyter, IPython is the default kernel defined as Kernel zero, which can be obtained through the kernel **ipykernel**. Jupyter supports the use of 80+ kernels and [3] shows how each of them can be installed, required dependencies and the correponding programming language version needed.

3.2. Interface

Zeppelin leverages a common UI framework with Bootstrap and Angular.js for the notebook interface. It is a easy to use interface with the support for creating dynamic forms within Zeppelin's notebook for which input can be obtained from a programming languages' output. Zeppelin provides an interface that is similar to RShiny's interactive web interface which allows user to manipulate in the front end rendered using Javascript with R running in the background [8].

Jupyter, on the other hand has a simple interface that is not user interactive as Zeppelin. When jupyter notebook is launched, the first page that is encountered is the Notebook Dashboard which shows the notebook files that has been created already and residing in the server. A new notebook can be created that is associated to a specific programming language's kernel (Python 2/Python 3/ R etc.) [9]. It's interface simple interface having a cell where the code can be written. Once the code is executed, the area below the cell would display the corresponding output.

3.3. Supported Languages

Zeppelin has the support for the 16 interpreters mentioned in Table 1. Since Apache Zeppelin has the default interpreter as spark which is a one of the major technology used for solving big data problems, the list includes interpreters like hbase, cassandra, elastic search etc. that works along with spark in solving big data problems.To overcome the limitation of these list of interpreters, Zeppelin provides the support for writing our own interpreter as described in [10].

Table 1. Programming languages supported by Apache Zeppelin

alluxio	file	jdbc	md
angular	flink	kylin	postgresql
cassandra	hbase	lens	python
elasticsearch	ignite	livy	shell

Jupyter, on the other hand has a huge list of about 80+ kernels being supported currently [3]. Though Jupyter has an upper-hand over Zeppelin over the number of supported programming languages, it lags behind Zeppelin in the support of using different programming language in the same notebook.

3.4. Visualization

Zeppelin provides the freedom to play around with various types of chart as shown in figure 2. It provides charting options by default for the output generated as a result of execution of code. The Apache Zeppelin community has been working on Project Helium [11], which aims to seed growth in all kinds of visualizations. This follows the model created by pluggable interpreters. Helium aims to make adding a new visualization simple, intuitive and can be accessed through a packaged code.

Jupyter, on the other hand has no charting options by default. Hence, it relies on existing charting libraries from the programming language that the notebook uses.

3.5. Multi-user capability

Zeppelin is still working towards providing multi user capability. Jupyter Hub [12] provides multiple user support for Jupyter. It

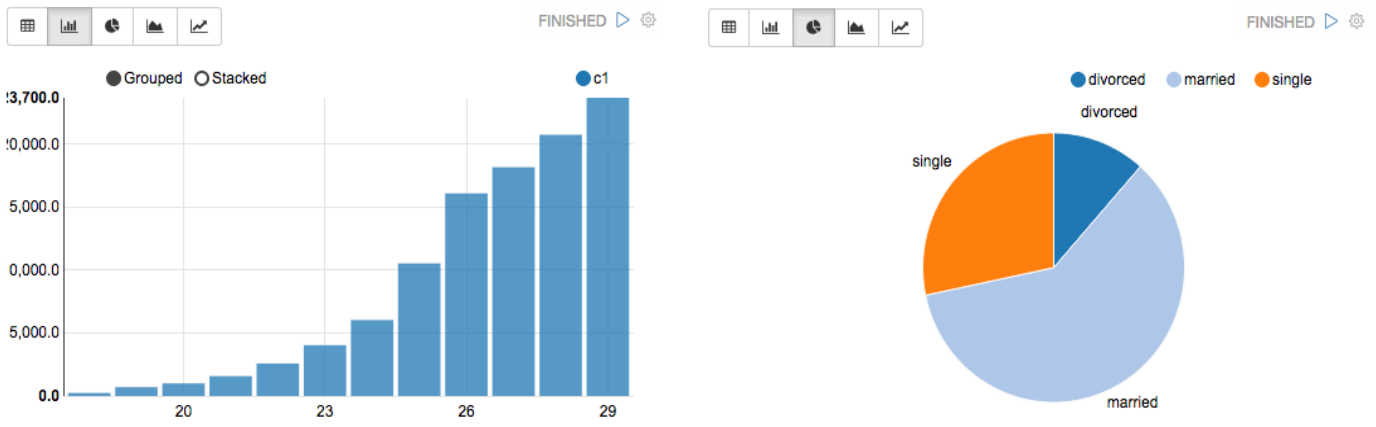


Fig. 2. Snapshot of Integrated graph - Apache Zeppelin. Different types of plots that can be readily viewed for the generated outputs in Zeppelin Notebook [5]

simple to setup because it leverages the Linux users and groups to provide authentication.Three subsystems make up JupyterHub as shown in Figure 3. Each user cannot touch or see any other users because it's restricted at the OS layer. However, Jupyter Hub has to be maintained in a single server which would provide access to multiple users.

1. a multi-user Hub (tornado process)
2. a configurable http proxy (node-http-proxy)
3. multiple single-user Jupyter notebook servers (Python/IPython/tornado)

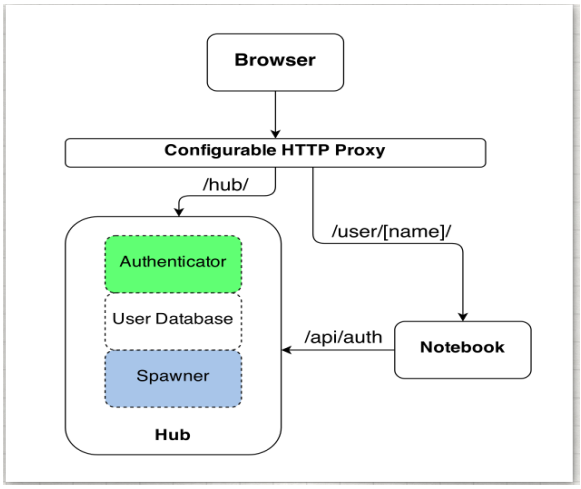


Fig. 3. Jupyter HUB Architecture.It allows for multiple users to interact with the Notebook server to access their respective Notebooks by means of user side authentication.[12]

3.6. Community support

Zeppelin is still in the incubation stage of Apache. It is progressing very slowly relatively to the status of Jupiter today. However, since Jupyter has a long history as IPython, there is lot of support for Jupyter in the online communtiy. A simple google search for the term **IPython** gives around 1.3 million results whereas for **Apache Zeppelin**, it is around 0.45 million. Apache Zeppelin is working with NFLabs, Twitter, Hortonworks, MapR, Pivotal, and IBM among many others delivering new features and fix issues in its platform [13].

4. ALTERNATIVES TO JUPYTER AND APACHE ZEPPELIN

4.1. Beaker Notebook

The Beaker Notebook is built on top of IPython kernel and was designed from the start to be a fully polyglot notebook [14]. It currently supports Python, Python3, R, Julia, JavaScript, SQL, Java, Clojure, HTML5, Node.js, C++, LaTeX, Ruby, Scala, Groovy, Kdb. It follows a cell type interface similar to Jupyter and Zeppelin and allows the user to use multiple programming languages across different cells in the same notebook. (i.e.) For example, the output of the code written in R in cell 1 can be accessed by a block of code written in Python in cell 2 for further manipulations.

4.2. SageMath

Sagemath is a is a free open-source mathematics software system [15] built for creating an open source alternative to Magma, Maple, Mathematica, and MATLAB. It is build on top of existing open-source packages: NumPy, SciPy, matplotlib, Sympy, Maxima, GAP, FLINT, R etc. It also offers a notebook type interface and the Sage Notebook recently moved to the cloud with SageMathCloud in collaboration with Google's cloud services.

5. CONCLUSION

The very need for experiments, explorations, and collaborations in scientific programming community is addressed by the evolution of these notebooks.Considering these into account, Apache Zeppelin is gaining upper hand over Jupyter in providing a fluid environment to solve big data problems apart from it being in the initial stages of defining multi-user support and relatively small community. With Zeppelin being a part of Apache community, it would be understandable that there would be constant

growth and updates. This can be strengthened from the fact that Apache Zeppelin is in the initial stages of developing Helium which would make visualization easy to use for big data problems.

REFERENCES

[1] Wikipedia, "Jupyter -debian wiki," dec 2016, [Online; accessed 23-March-2017]. [Online]. Available: https://wiki.debian.org/Jupyter

[2] Jupyter, "What is the jupyter notebook?" Web Page, jan 2017, accessed: 2017-03-02. [Online]. Available: http://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html

[3] Jupyter, "Jupyter kernels," Web Page, feb 2017, accessed: 2017-03-02. [Online]. Available: https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

[4] Jupyter, "How ipython and jupyter notebook work," Web Page, feb 2017, accessed: 2017-03-02. [Online]. Available: http://jupyter.readthedocs.io/en/latest/architecture/how_jupyter_ipython_work.html

[5] Apache, "Apache zeppelin," Web Page, nov 2016, accessed: 2017-03-02. [Online]. Available: http://zeppelin.apache.org/

[6] Apache, "Interpreters in apache zeppelin," Web Page, nov 2016, accessed: 2017-03-02. [Online]. Available: http://zeppelin.apache.org/docs/latest/manual/interpreters.html

[7] Apache, "Interpreters installation," Web Page, nov 2016, accessed: 2017-03-02. [Online]. Available: https://zeppelin.apache.org/docs/0.6.0/manual/interpreterinstallation.html

[8] RStudio, "Shiny," Web Page, dec 2016, accessed: 2017-03-22. [Online]. Available: https://shiny.rstudio.com/

[9] Jupyter, "Ui components - jupyter notebook," Web Page, feb 2017, accessed: 2017-03-02. [Online]. Available: http://jupyter-notebook.readthedocs.io/en/latest/ui_components.html

[10] Apache, "Writing a new interpreter," Web Page, nov 2016, accessed: 2017-03-02. [Online]. Available: http://zeppelin.apache.org/docs/latest/development/writingzeppelininterpreter.html#make-your-own-interpreter

[11] Wikipedia, "Helium proposal," mar 2016, [Online; accessed 21-March-2017]. [Online]. Available: https://cwiki.apache.org/confluence/display/ZEPPELIN/Helium+proposal

[12] Jupyter, "Jupyterhub," Web Page, feb 2017, accessed: 2017-03-22. [Online]. Available: https://jupyterhub.readthedocs.io/en/latest/

[13] hortonworks, "Jupyterhub," Web Page, jun 2016, accessed: 2017-03-23. [Online]. Available: https://hortonworks.com/blog/apache-zeppelin-road-ahead/

[14] T. Sigma, "Beaker notebook," Web Page, dec 2016, accessed: 2017-03-25. [Online]. Available: http://beakernotebook.com/

[15] S. Foundation, "Beaker notebook," Web Page, jan 2017, accessed: 2017-03-25. [Online]. Available: http://www.sagemath.org/

Introduction to Terraform

SUSHMITA SIVAPRASAD[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: sushsiva@umail.iu.edu*

project-000, April 30, 2017

This paper gives a brief introduction on Terraform and how infrastructure can be used as a code, which is the building block of Terraform. It is written in go scripting code and it is a server provisioning tool where we can specify what is the goal we require and Terraform creates steps of tasks on how to reach the goal. This paper provides information on what are the use cases of Terraform and how it differentiates itself from other tools. Resources for learning Terraform in much more detail has been provided as well.

Keywords: Terraform, IAC, Go code, Ansible

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IR-2038/report.pdf

1. INTRODUCTION

Terraform is an open source tool created by HashiCorp. It is an infrastructure management tool written in Go programming language.It allows users to safely and predictably create , change and improve the production infrastructure and codifies the API's into a declarative configuration file that can be shared among other users as well[1]. The tool can even be treated as a code whereby we can edit, review and create versions at the same time. The tool is popular in allowing users to create a customized in-house solution. It allows users to build an executive plan which can be used for the purpose of creating applications and implementing the infrastructure[2].

2. INFRASTRUCTURE AS A CODE

Infrastructure as a code or IAC allows us to write and execute a code in order to define, deploy and update the infrastructure. There are 4 categories of the IAC tools,

2.1. Ad hoc scripts

Ad Hoc scripts allows to automate anything. We can use any scripting language and break down the tasks we were doing manually and execute the script on the server. Ad hoc script allows one to write the code in the required manner[3].

2.2. Configuration management tools

Some of the main configuration management tools are Chef, Ansible, SaltStack which are designed to install and manage software on any of the existing servers. These tools are designed to manage large number of remote servers. Ansible playbook can be used to configure multiple servers in a parallel mode[3]. A parameter called serial can be set in the playbook from which a rolling deployment can be done and the server will be updated batch wise. Tools such as Ansible are idempotent, that is they run correctly no matter how many times we run the code[3].

2.3. Server templating tools

Server templating tools are Docker, Packer and Vagrant. A server templating tool allows to create an image of a server that captures a snapshot image of the operating system, the software and the files in it. The image of the server can then be distributed across all of the servers using Ansible[3].

2.4. Server provisioning tools

Server provisioning tools includes Terraform, Cloudformation, Openstack Heat[3]. These tools allows to create a server by themselves. These tools can also be used to create databases, caches, load balancers, queues, firewall settings and other aspects from the infrastructure[3].

3. WORKING OF TERRAFORM

Infrastructure is the building block on which the application is created. Terraform provides us a declarative execution plan for building and running applications and infrastructure. All that we are required to do in a declarative state is we declare the required end of the state. The tool will take care of the steps required to reach the goal state. It allows the user to not be bothered about the commands to run or the settings required to change. The user has to declare the resources using a graph-based approach in order to model and apply the desired state[2]. The Go code allows Terraform to compile down into a single binary code for each of the supported operating systems. This binary is used to deploy infrastructure from our pc or a server and we would not require an additional infrastructure to make

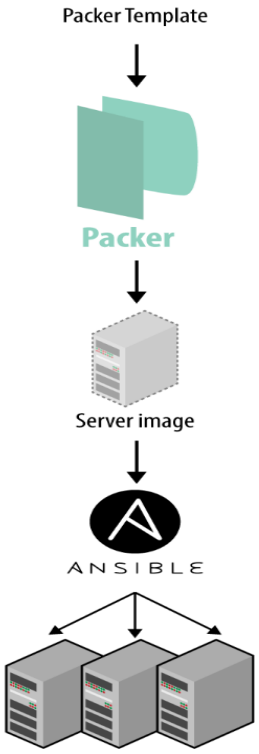


Fig. 1. Packer is used to create an image of a server which can be installed using Ansible across all other servers [3]

that happen. The binary makes API calls on behalf of us to one or more of the providers such as Azure, AWS, Google Cloud, etc. In this way terraform lets users to use the infrastructures provided by these providers as well as the authentication mechanism we are already using with these providers [3]. Terraform allows users to deploy interconnected resources across various cloud providers at the same time. It translates the contents of the configurations into API calls into the cloud providers as shown in figure 2.

4. USE CASES

4.1. Multi-Tier Application

It is useful for building multi-tier applicatin which consists of web, cache, application, middleware and tiers of database. Terraform helps to build N-tier applications. Each of the tier can be configured in Terraform and left independent and isolated which can be easily scaled and managed by the code[2].

4.2. Disposable Environments

In order to test a new application before it has been sent for production , a staging or QA Environment is required in order to avoid complexity and confusion. For such a situation terraform can be very handy, where the production environment can be codified and shared along with staging[2]. It can even create new environment for it to test on and which can be later disposed of , keeping only what is required. Terraform makes it easy to maintain paralle environements and easily dispose them of [2].

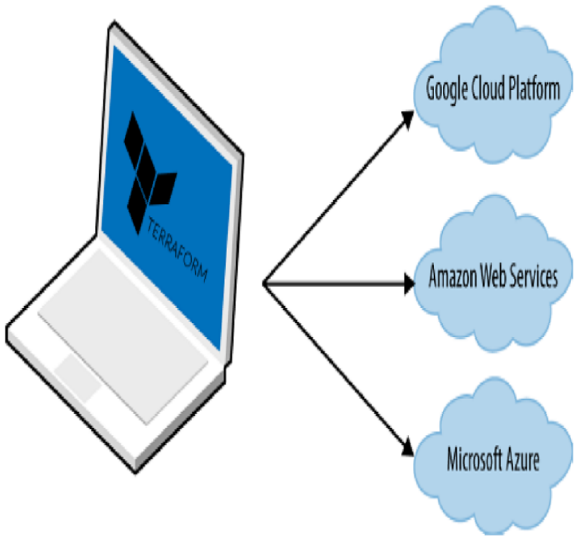


Fig. 2. Figure depicting terraform transforming the configurations to API calls into the cloud providers [2]

4.3. Multi-Cloud Deployment

In order to increase the fault tolerance, infrastructure is spread across multiple clouds to continue it's progress irrespective of any faults. The current multi-cloud deployement consists of cloud specific tool for infrastructure management. Terraform allows a single configuration to be used across multiple providers and also handle any cross-cloud dependencies[1].

5. ADVANTAGES OF USING TERRAFORM

Terraform allows the existing tools to focus on its strength such as bootstrapping and initializing resources. It makes the infrastructure deployment easy by focusing on a higher level of abstraction of the datacenter and also allowing the same codification for the tools. It is cloud agnostic and allows multiple providers and services to be combined [4]. It separates the planning and execution phase. It generates an action plan when we provide a goal state, which keeps getting updated when we add or remove new resources. The terraform graph feature allows the user to visualize the plan in order for them to know exactly the effects of the changes before they get implemented [4].

6. EDUCATIONAL MATERIAL

Hashicorp has provided a documentation for describing Terraform, it's download and installation procedure. There has been 2 books published by authors Yevgeniy Brikman[3] and James Turnbull[2] which gives a very detailed insight about the working and usage of Terraform.

7. CONCLUSION

Terraform simplifies the process of creating applications and implementing the infrastructure by just specifying the required goal. It uses Go code which allows Terraform to compile down into a single binary code for each of the supported operating systems. We can create N- tier applications with ease by just

providing the resources and the end state of the required application. It allows multiple user to work on a cloud-agnostic platform thus being a very versatile tool.

8. ACKNOWLEDGEMENT

A very special thanks to Professor Gregor von Laszewski and the teaching assistants Miao Zhang and Dimitar Nikolov for all the support and guidance in getting this paper done and resolving all the technical issues faced. The paper is written during the spring 2017 semester course I524: Big Data and Open Source Software Projects at Indiana University Bloomington.

9. AUTHOR BIOGRAPHY

Sushmita Sivaprasad is a graduate student in Data Science at Indiana University under the department of Informatics and Computing. She had completed her bachelors in Electronics and Communication from SRM University , India and her master's in International Business from Hult International Business School, UAE.

REFERENCES

[1] "Introduction to terraform," Web Page, Oct. 2014, accessed: 2017-03-25. [Online]. Available: https://www.terraform.io/intro/index.html

[2] J. Turnbull, *The Terraform Book*, 110th ed. Turnbull Press, Nov. 2016, accessed:2017-03-25.

[3] Yevgeniy Brikman, *Terraform: Up and Running:Writing Infrastructure as Code*, 1st ed. O'reilly media, Mar. 2017, accessed : 2017-2-24. [Online]. Available: http://shop.oreilly.com/product/0636920061939.do

[4] "Terraform vs. other software," Web Page, Oct. 2014, accessed: 2017-03-27. [Online]. Available: https://www.terraform.io/intro/vs/cloudformation.html

Google BigQuery - A data warehouse for large-scale data analytics

SAGAR VORA[1]

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

April 30, 2017

The amount of relational data generated is increasing rapidly since it is generated through a large number of sources. Moreover this data is the information which companies like to explore and analyse quickly to identify solutions to business. Therefore, the need to solve the problem of traditional database management systems in order to support large volumes of data arises Google's BigQuery platform. Google BigQuery is an enterprise data warehouse used for large scale data analytics. A user can store and query the massive datasets by storing data in BigQuery and quering the database. BigQuery runs in cloud using the processing power provided by Google's infrastructure and provides SQL-like queries to perform analysis on masssive quantities of data, providing real-time insights about the data.

Keywords: BigQuery, BigData, Google, Cloud, Database, IaaS, PaaS, SaaS, SQL

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IR-2041/report.pdf

1. INTRODUCTION

Nowadays, the amount of data being collected, stored and processed continues to grow rapidly. Querying this massive datasets can be time consuming and expensive without the right hardware and infrastructure. BigQuery [1] solves this problem by providing super-fast, SQL-like queries, using the processing power of Google's infrastructure. Google BigQuery [2] is a cloud web service data warehouse used for large-scale data processing. It is suitable for businesses that cannot afford to spend a huge amount of investment in infrastructure to process a huge amount of information. This platform allows to store and retrieve large amounts of information in near real time as well as providing some important analysis of the data which is stored.

2. GOOGLE BIGQUERY

To solve the architectural problems faced by Hadoop [3] MapReduce [4], Google developed Dremel [5] application in order to process large volumes of data. Dremel was designed to deliver high performance on data which was spread across a huge range of multiple servers and SQL support. But in 2012 at Google I/O event, it was announced that they would no longer support Dremel and this led to the beginning of BigQuery which became then the high-performance cloud offering of google. BigQuery makes use of SSL (Secure Sockets Layer) as well to take care of the security concerns related to cloud management.

2.1. System Architecture and Internal Structure

Google BigQuery platform is a Software As a Solution (SaaS) model in the cloud. As seen in figure 1, data which is generated from a variety of sources like event logs, relational databases, IoT devices like sensors, actuators, social media websites, Informatica [6] applications, etc is loaded in the databases which are created in BigQuery.

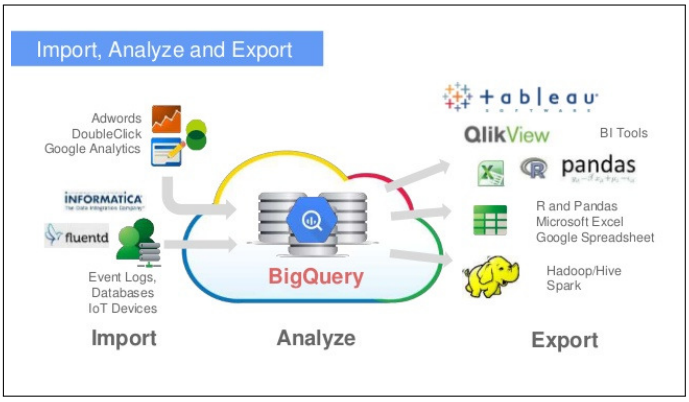


Fig. 1. [7] System Architecture of BigQuery

This data can then be processed and anaylsed using some algorthmic logic or using some processing tool. Finally the data can then be represented and exported using Tableau [8], Qlikview [9], MS Excel [10] and other BI tools. It can also be exported on

Hadoop system for parallel processing.
Now to store data in BigQuery, you need to create Projects. Projects [11] in BigQuery act as top-level containers which store the BigQuery data. Each project is referenced by a name and unique ID. Tables in BigQuery store the actual data where each table has a schema which describes the field name, types and other information. In BigQuery each table must belong to a dataset. Datasets help to organise the tables and control the access to it.

3. FEATURES OF BIGQUERY

Google BigQuery presents some characteristics like velocity, simplicity, and multiple access methods.

3.1. Velocity

BigQuery can process millions of information in seconds because of its columnar storage, and tree-like architecture.

3.2. Columnar Storage and Tree Architecture

The data instead of being stored in terms of rows like in standard SQL, is stored as columns and thus storage is oriented as shown in figure 2. This not only results in fast access to data but also results in scanning only the required values, which largely reduces latency. This also results in a higher compression ratios. Google reports [12] that "BigQuery can achieve columnar compression ratios of 1:10 as opposed to 1:3 when storing data in a traditional row-based format".

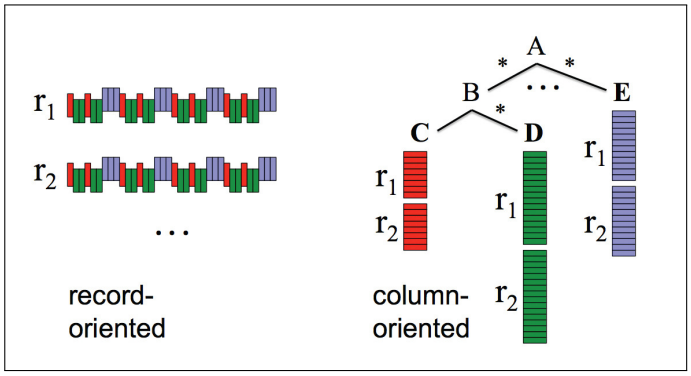


Fig. 2. [12] Row-oriented vs Column-oriented Storage in tree architecture

Moreover the tree-like architecture is used for processing queries and aggregating results across variety of different nodes. The tree-like structure also helps in retrieving data faster.
Let us see this with the help of 2 examples.
Considering the column-oriented storage in tree architecture format in the figure 2, lets refer node A as our root server, node B being an intermediate server and C, D, E being leaf servers having local disk storage space.

Example 1: Fast-retrieval
Statement: Find out all the customer names whose name starts with 'A'.
Assuming node C contains customer names. Hence, it is as simple as traversing A -> B -> C and looking at the datasets Cr1 and Cr2 for names starting with 'A'. One need not look at the paths from A -> B -> D and A -> E. Hence, in this simple scenario, A query may not scan the entire storage structure of the BigQuery and hence speeds up retrieving the information. Here the reason why the query looks only in the path A -> B -> C because BigQuery knows that all the customer names starting with A have been placed in the local disk storage at node C itself.

Example 2: Parallel Processing
Statement: Count all the customer whose names starting with 'A'.

1. Root node A sends out the query to node B which in turn translates the query to sum the names of all the customers starting with 'A'.
2. Now after translating, node B passes this query to leaf node C which has data stored in columnar format in the form of r1 and r2 tables.
3. node C accesses these tablets in parallel, and counts the number of customers whose names start with 'A' and passes the count of Cr1 and Cr2 to node B which sums the result and passes it to the root Aas the output of the query.

So this not only increases the speed of retriving the information but also the goal of parallel processing which is the need in quering huge datasets is also achieved.

3.3. Simplicity

Figure 3 shows the simple user interface provided by BigQuery. The dataset stored in BigQuery can be easily queried using SQL-like queries.

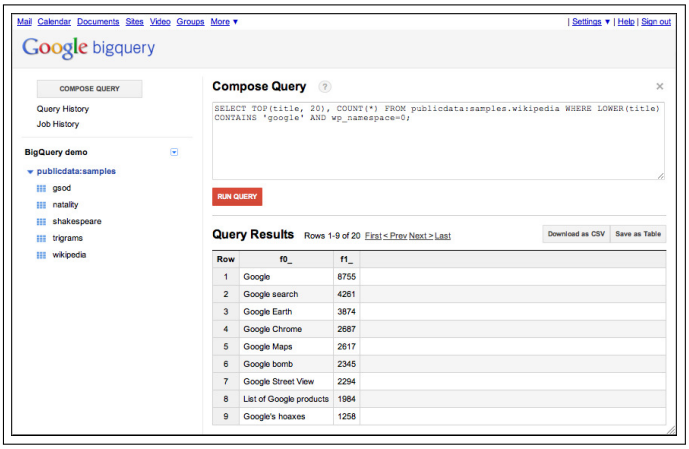


Fig. 3. [13] Big Query Sample User Interface

You can enter your query in the text area provided and the results will be displayed below. At the left side of the interface, it lists all the tables related to a particular database which are referenced as projects in BigQuery. In figure 3, the left-hand side consists a list of all the tables related to the 'publicdata:samples' project. The query is fired in the text area below the 'Compose Query' text and its results are displayed below in tabluar format.

3.4. Multiple access methods

You can access the BigQuery service in 3 different ways. We can either use a BigQuery browser tool, a bq command-line tool or a REST-based API.

1. BigQuery Browser Tool: It allows to easily browse, create tables, run queries, and export data to Google Cloud Storage.

2. bq command-line Tool: This is a Python command - line tool which permits to manage and query the data.

3. REST API: We can access BigQueryeven by making calls to the REST API using a variety of client libraries such as Java, PHP or Python.

4. COMPARISON OF BIGQUERY WITH IMPALA, SHARK, HIVE, REDSHIFT AND TEZ

In [14], the performance [15] of BigQuery is compared against Impala [16], Shark [17], Hive [18], Redshift [19] and Tez [20] by using Intel's Hadoop benchmark [21] tools . This benchmark has 3 different sizes of datasets - tiny, 1node and 5nodes. The *'rankings'* table has 90 million records and *'uservisits'* table has 775 million records. The data is taken using Common Crawl [22] document corpus. The two tables have the following schemas:

Rankings table schema: (lists websites and their page rank)
- pageURL (String)
- pageRank (Integer)
- avgDuration (Integer)

Uservisits table schema: (Stores server logs for each web page)
- sourceIP (String)
- destURL (String)
- visitDate (String)
- adRevenue (Float)
- userAgent (String)
- countryCode (String)
- languageCode (String)
- searchWord (String)
- duration (Integer)

The benchmark measures response time on 3 different types of queries. One being a basic scan query, one being an aggregation query, and one join query.

1. Scan Query

   Figure 4 shows a general scan query on the rankings table.

```
SELECT
    pageURL,
    pageRank
FROM
    benchmark.rankings
WHERE
    pageRank > X
```

Fig. 4. [14] Scan Query

This query has been fired using different values for 'X'. Query 1A means X's value is 1000, query 1B means X's value is 100 and 1C means X's value is 10.

2. Aggregation Query

   Figure 5 shows an aggregation query on the uservisits table. The aggregation is performed using the sum function along with the substr function.

```
SELECT
    SUBSTR(
        sourceIP, 1, X)
AS srcIP,
    SUM(adRevenue)
FROM
    benchmark.uservisits
GROUP EACH BY
    srcIP
```

Fig. 5. [14] Aggregation Query

This query has been fired using different values for 'X'. Query 2A means X's value is 8, query 2B means X's value is 10 and 2C means X's value is 12.

3. Join Query

   Figure 6 shows a join query between the rankings table and uservisits table on some given condition.

```
SELECT
    sourceIP,
    sum(adRevenue) AS totalRevenue,
    avg(pageRank) AS pageRank
FROM
    benchmark.rankings R
JOIN EACH(
        SELECT
            sourceIP,
            destURL,
            adRevenue
        FROM
            benchmark.uservisits UV
        WHERE
            UV.visitDate > "1980-01-01"
            AND
            UV.visitDate < X
        ) NUV
ON
    (R.pageURL = NUV.destURL)
GROUP EACH BY
    sourceIP
ORDER BY
    totalRevenue DESC LIMIT 1
```

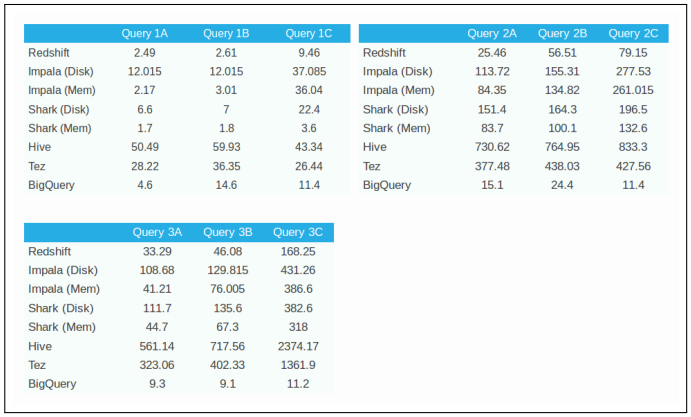
Fig. 6. [14] Join Query

Like other queries earlier, this one also has different values

for 'X'. Query 3A means X's value is '1980-04-01', query 3B means X is '1983-01-01' and 3C means X is '2010-01-01'.

4.1. Comparison results

Figure 7 shows the results.



	Query 1A	Query 1B	Query 1C
Redshift	2.49	2.61	9.46
Impala (Disk)	12.015	12.015	37.085
Impala (Mem)	2.17	3.01	36.04
Shark (Disk)	6.6	7	22.4
Shark (Mem)	1.7	1.8	3.6
Hive	50.49	59.93	43.34
Tez	28.22	36.35	26.44
BigQuery	4.6	14.6	11.4

	Query 2A	Query 2B	Query 2C
Redshift	25.46	56.51	79.15
Impala (Disk)	113.72	155.31	277.53
Impala (Mem)	84.35	134.82	261.015
Shark (Disk)	151.4	164.3	196.5
Shark (Mem)	83.7	100.1	132.6
Hive	730.62	764.95	833.3
Tez	377.48	438.03	427.56
BigQuery	15.1	24.4	11.4

	Query 3A	Query 3B	Query 3C
Redshift	33.29	46.08	168.25
Impala (Disk)	108.68	129.815	431.26
Impala (Mem)	41.21	76.005	386.6
Shark (Disk)	111.7	135.6	382.6
Shark (Mem)	44.7	67.3	318
Hive	561.14	717.56	2374.17
Tez	323.06	402.33	1361.9
BigQuery	9.3	9.1	11.2

Fig. 7. [14] Comparison of BigQuery with other storage platforms

In this comparison experiment, each query was executed atleast 10 times and the results which are displayed are the average values of the response time in seconds. From 7, it shows that only in Query 1A, 1B and 1C BigQuery took more time than Redshift, Impala (on memory) and Shark (on memory) but the results related to query 2 and 3 are much better. For all the different values of X in query 2 and 3 (A,B and C), BigQuery executed them faster than other platforms. This shows that BigQuery can produce better results when processing complex queries on large datasets. As the number of processing records grew, BigQuery's response time was less as compared to the other storage platforms.

5. USE CASES OF BIGQUERY

5.1. Safari Books Online

Safari Books Online [23] uses BigQuery to find trends in customer purchase, manage its negative feedback related to customer service, improve the sales team's effectiveness etc. They chose BigQuery over other technologies because of its retrieval speed by quering the datasets using a familiar SQL-like language, and the lack of additional required maintenance.

5.2. RedBus

Online travel agency RedBus [24] introduced internet bus ticketing in India incorporating thousands of bus schedules into a single booking operation. Using BigQuery, redBus was able to manage the terabytes of booking and inventory data quickly and at a lower cost than other big-data services. BigQuery also helped its engineers fix the software bugs quickly, minimize the lost sales and improve its customer service as well.

6. CONCLUSION

Google BigQuery is a cloud-based database service which enables to process large data sets quickly. BigQuery allows to run SQL-like queries against multiple gigabytes to terabytes of data in a matter of seconds. It is suitable for ad-hoc OLAP/BI [25] queries that require results as fast as possible. As a cloud-powered parallel query database it provides extremely high full-scan query performance and cost effectiveness compared to other traditional data warehouse solutions and appliances.

ACKNOWLEDGEMENTS

I would like to thank my professor Gregor von Laszewski and all the associate instructors for their constant technical support.

REFERENCES

[1] "What is bigquery," Web Page, accessed: 2017-3-24. [Online]. Available: https://cloud.google.com/bigquery/

[2] S. Fernandes and J. Bernardino, "What is bigquery?" in *Proceedings of the 19th International Database Engineering & Applications Symposium*. New York, NY, USA: ACM, 2014, pp. 202–203. [Online]. Available: http://doi.acm.org.proxyiub.uits.iu.edu/10.1145/2790755.2790797

[3] "Apache hadoop," Web Page, accessed: 2017-3-25. [Online]. Available: http://hadoop.apache.org/

[4] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008, published as an Article. [Online]. Available: http://doi.acm.org/10.1145/1327452.1327492

[5] 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*, Sep. 2010, pp. 330–339. [Online]. Available: http://www.vldb2010.org/accept.htm

[6] "What is informatica," Webpage, accessed: 2017-4-07. [Online]. Available: https://www.informatica.com

[7] K. Sato, "Fluentd + google bigquery," Web Page, Mar. 2014, accessed: 2017-3-24. [Online]. Available: https://www.slideshare.net/GoogleCloudPlatformJP/google-for-1600-kpi-fluentd-google-big-query

[8] "What is tableau," Webpage, accessed: 2017-4-07. [Online]. Available: https://www.tableau.com/

[9] "Qlik," Webpage, accessed: 2017-4-07. [Online]. Available: http://www.qlik.com/us/

[10] "Microsoft excel," Webpage, accessed: 2017-4-07. [Online]. Available: https://products.office.com/en-us/excel

[11] "What is bigquery? bigquery documentation google cloud platform," Web Page, accessed: 2017-3-23. [Online]. Available: https://cloud.google.com/bigquery/what-is-bigquery

[12] V. Agrawal, "Google bigquery vs mapreduce vs powerdrill," Web Page, accessed: 2017-3-23. [Online]. Available: http://geeksmirage.com/google-bigquery-vs-mapreduce-vs-powerdrill

[13] J.-k. Kwek, "Google bigquery service: Big data analytics at google speed," Blog, Nov. 2011, accessed: 2017-3-23. [Online]. Available: https://cloud.googleblog.com/2011/11/google-bigquery-service-big-data.html

[14] V. Solovey, "Google bigquery benchmark," Blog, Jun. 2015, accessed: 2017-3-23. [Online]. Available: https://www.doit-intl.com/blog/2015/6/9/bigquery-benchmark

[15] "Amp lab big data benchmark," Webpage, accessed: 2017-4-08. [Online]. Available: https://amplab.cs.berkeley.edu/benchmark/

[16] M. Kornacker and J. Erickson, "Cloudera impala: Real-time queries in apache hadoop, for real," Blog, Oct. 2012, accessed: 2017-4-07. [Online]. Available: http://blog.cloudera.com/blog/2012/10/cloudera-impala-real-time-queries-in-apache-hadoop-for-real/

[17] "Apache spark," Webpage, accessed: 2017-4-07. [Online]. Available: https://spark.apache.org/sql/

[18] "Apache hive," Webpage, accessed: 2017-4-07. [Online]. Available: http://hive.apache.org/

[19] "Amazon's redshift," Webpage, accessed: 2017-4-07. [Online]. Available: https://aws.amazon.com/redshift/

[20] C. Shanklin, "Announcing stinger phase 3 technical preview," Blog, Dec. 2013, accessed: 2017-4-08. [Online]. Available: https://hortonworks.com/blog/announcing-stinger-phase-3-technical-preview/

[21] "Intel's hadoop benchmark," Git Repo, accessed: 2017-4-07. [Online]. Available: https://github.com/intel-hadoop/HiBench
[22] "Common crawl," Webpage, accessed: 2017-4-07. [Online]. Available: http://commoncrawl.org/
[23] D. Peter, "How safari books online uses bigquery for business intelligence," Webpage, accessed: 2017-4-08. [Online]. Available: https://cloud.google.com/bigquery/case-studies/safari-books
[24] "Travel agency masters big data with google bigquery," Webpage, accessed: 2017-4-08. [Online]. Available: https://cloud.google.com/customers/redbus/
[25] "OLAP," Webpage, accessed: 2017-4-08. [Online]. Available: http://olap.com/olap-definition/

Hive

DIKSHA YADAV[1,*,+]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: yadavd@umail.iu.edu*
[+] *HID - S17-IR-2044*

Paper-002, April 30, 2017

Hive is an open source data warehousing solution which is built on top of Hadoop. It structures data into understandable and conventional database terms like tables, columns, rows and partitions. It supports HiveQL queries which have structure like SQL queries. HiveQL queries are compiled to map reduce jobs which are then executed by Hadoop. Hive also contains Metastore which includes schemas and statistics which is useful in query compilation, optimization and data exploration.

Keywords: Hive, Hadoop, HiveQL, SQL, HDFC, RDBMS

https://github.com/cloudmesh/sp17-i524/tree/master/paper2/S17-IR-2044/report.pdf

1. INTRODUCTION

Hive is an ETL and open source data warehousing solution which is built on top of Hadoop Distributed File System. Hive was built in January 2007 and open sourced in August 2008. It structures data into understandable and conventional database terms like tables, columns, rows and partitions. It supports HiveQL queries which have structure like SQL queries. HiveQL queries are compiled to map reduce jobs which are then executed by Hadoop. Hive also contains Metastore which includes schemas and statistics which is useful in query compilation, optimization and data exploration. In short, hive can be used by analyzing huge datasets, performing encapsulation of data and running ad hoc queries [1]

2. ARCHITECTURE

Hive architecture as provided in Fig.1. from [2] has, Database-It consists of tables created by the user. Hadoop Distributed File System and or Hbase are used as data storage techniques to store data in file system. Metastore-It contains information about the system. It can be accessed by different components as and when needed. All components of hive interact with metastore Interfaces-User interface and Application programming interface both are present in hive. External interfaces which includes Command Line Interface Web User Interface. Also it contains JDBC and ODBC Application programming Interfaces Driver-manage HiveQL statements at every stage which includes compilation stage, optimization stage and execution stage. A session handle is created every time a Hive QL statement is received from any interfaces or thrift server which records information like number of output rows , execution time etc. Query compiler-It compiles HiveQL queries to acyclic graphs (directed) representing map reduce tasks. Execution Engine- It executes the tasks generated by the compiler. Hive Server- It provide JDBC/ODBC server and thrift interface. Compiler-When a Hive QL statement is received from interface, the driver invokes the compiler for performing its task of translating the Hive QL statement into Directed acyclic graph of map reduce jobs. The map reduce jobs are then submitted by the driver to execution engine (like Hadoop) in a topological order [2]

3. QUERY EXECUTION IN HIVE

When a query is submitted to the hive. Compiler compiles the query. The compiled query is executed by execution engine like MapReduce. Resources are then allocated across the clusters for application by the resource manager, YARN. The data that is used by the query is stored in HDFS (Hadoop Distributed File System). Supported data formats are AVRO, Parquet, ORC and text.
When results from query are ready, they are set back using JDBC/ODBC connection. [3]

4. FEATURES

Hive can be used with structured or semi structured data only.
HiveQL does not require user to deal with MapReduce complex programming.
Infact user has to use concepts similar to relational database like tables, schema, rows, columns etc.
Hive supports four file formats, text file, sequence file, orc and rcfile.
HiveQL syntax is similar to SQL syntax.

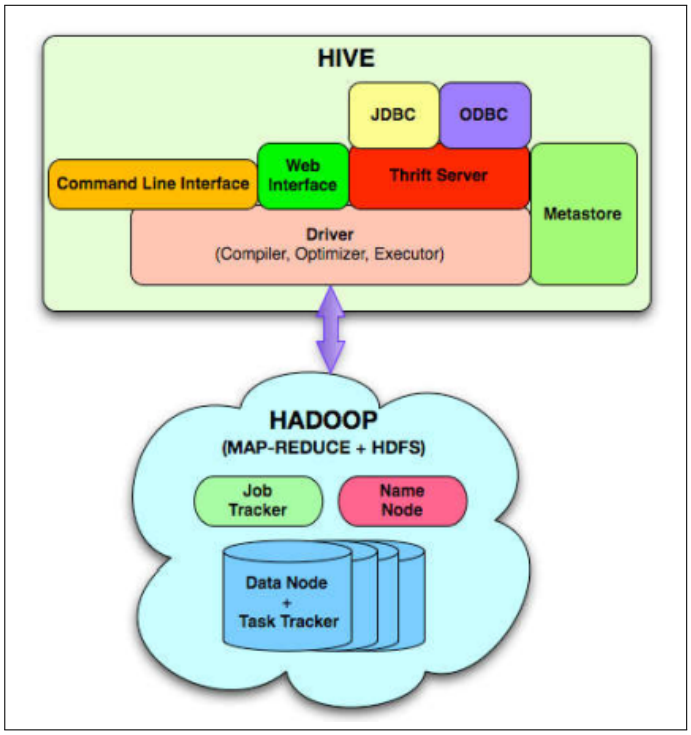


Fig. 1. Hive Architecture

Hive query executes on Hadoop's infrastructure rather than traditional database.
Hive uses partition concept for data retrieval.
Hive supports custom User defined functions for data cleansing, filtering etc.
Hive can be used in two modes, local mode and MapReduce mode.
Selection of mode depends on certain conditions like data size, data nodes in Hadoop.
By default, hive runs on MapReduce mode.
[4]

5. SYSTEM REQUIREMENTS

Hive is cross platform. So, It does not need any specific operating system to work.
Minimum System Requirements:
CPU Speed:Intel Dual-Core 2.4 GHz or AMD equivalent
RAM:2 GB
OS:Windows 7
Video Card:NVIDIA GeForce 8800GT
Sound Card:DirectX®-compatible
Free Disk Space:1 GB
Prefered System Requirements:
CPU Speed:Intel Dual-Core 2.4 GHz or AMD equivalent
RAM:4 GB
OS:Windows 7
Video Card:NVIDIA GeForce GTX 260
Sound Card:DirectX®-compatible
Free Disk Space:1 GB
[5]

6. COMPARISON OF HIVE WITH OTHER TRADITIONAL DATABASES

Traditional databases like RDBMS follow schema on write approach that is read and write many times while Hive follows schema on read approach that is write once and read many times.
In schema on write, databases checks at load time if the data follows the table representation given by user while in schema on read approach, it is checked at run time only. This saves the time for hive to load the data when traditional databases takes longer time.
Hive does not support record level updates like RDMS. For example, we cannot perform delete, update, insert etc at record level in hive like we can perform in RDBMS.
Hive does not support OLTP (Online Transaction Processing), it only supports OLAP (Online Analytical Processing) whereas RDBMS supports both OLTP and OLAP.
For dynamic data analysis, RDMS would be preferred if quick responses are needed. Hive is suitable for data warehousing applications, where analysis is done on static data and fast responses are not needed.
One more difference between hive and RDBMS is that hive is scalable and that too at low cost, while scalability comes at higher cost in RDBMS.[6]

7. POPULARITY OF HIVE

The popularity of hive is increasing with time. This can be proved by the following plot made by DB Engines Ranking. It ranks database management systems according to their status and popularity. Following plot shows popularity of hive with time.

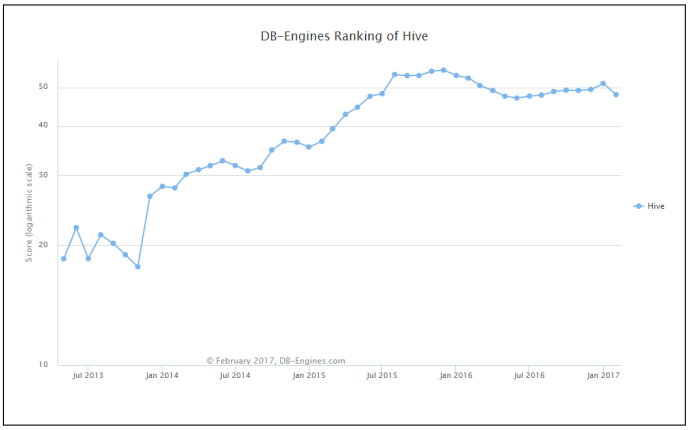


Fig. 2. Hive Popularity

[7]

8. RESOURCES FOR LEARNING HIVE

Someone new to hive can start learning it by going through the following links in sequence: Install Hivehttps://www.edureka.co/blog/apache-hive-installation-on-ubuntu?utm_source=quora&utm_medium=crosspost&utm_campaign=social-media-edureka-ab
Hive Tutorialhttps://www.edureka.co/blog/hive-tutorial/?utm_source=quora&utm_medium=crosspost&utm_campaign=social-media-edureka-ab

Top Hive commands with exampleshttps://www.edureka.co/blog/hive-commands-with-examples?utm_source=quora&utm_medium=crosspost&utm_campaign=social-media-edureka-ab

9. ACKNOWLEDGEMENT

I am also grateful to Dr. Gregor von Laszewski for providing the appropriate paper template.

10. CONCLUSION

Hive is an ETL and data warehouse tool on top of Hadoop framework which is used for processing structured and semi structured data.
Hive provides flexible query language such as HiveQL for querying and processing of data.
It makes working easier for the user as they do not have to deal with MapReduce programming complexity when using SQL like structure of HiveQL
It provides many new and better features compared to RDMS which has certain limitations.
It supports writing and deploying custom user defined scripts and User defined functions.

REFERENCES

[1] "Hive," Web Page, online; accessed 24-March-2017. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Hive
[2] "Hive article," Web Page, online; accessed 24-March-2017. [Online]. Available: https://docs.treasuredata.com/articles/hive
[3] "Hive architecture," Web Page, online; accessed 07-April-2017. [Online]. Available: https://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.4.3/bk_performance_tuning/content/ch_hive_architectural_overview.html
[4] "Hive," Web Page, online; accessed 07-April-2017. [Online]. Available: http://www.guru99.com/introduction-hive.html
[5] "System requirements hive," Web Page, online; accessed 07-April-2017. [Online]. Available: https://www.systemrequirementslab.com/cyri/requirements/the-hive/14593
[6] A. T. J. S. N. J. Z. S. P. C. S. A. H. L. P. W. R. Murthy, "Hive-a petabyte scale datawarehouse using hadoop," Paper. [Online]. Available: http://infolab.stanford.edu/~ragho/hive-icde2010.pdf
[7] "Ranking hive," Web Page, online; accessed 20-March-2017. [Online]. Available: http://db-engines.com/en/ranking_trend/system/Hive