**“APACHE HAMA”**

**-PRIYANSHU SHARMA**

**INTRODUCTION**

In today's highly intertwined network society, the demand for big data processing frameworks is continuously growing. The widely adopted model to process big data is parallel and distributed computing. This assignment documents the significant progress achieved in the field of distributed computing frameworks, particularly Apache Hama, a top level project under the Apache Software Foundation, based on bulk synchronous parallel processing.

Apache Hama is a distributed computing framework based on [bulk synchronous parallel](https://en.wikipedia.org/wiki/Bulk_synchronous_parallel" \o "Bulk synchronous parallel) computing techniques for massive scientific computations e.g., matrix, graph and network algorithms. It is a Top Level Project under the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation" \o "Apache Software Foundation). It was created by *Edward J. Yoon*, who named it (short for "Hadoop Matrix") and was inspired by [Google](https://en.wikipedia.org/wiki/Google" \o "Google)'s *Pregel* large-scale graph computing framework described in 2010

Apache Hama is written in Java. Hama provides a Pure BSP Bulk Synchronous Parallel Model for message passing and collective communication. A BSP program consists of a sequence of super-steps. BSP programming enables you to write high-performance parallel computing algorithms for a wide range of scientific problems.It serves a purpose similar to the [parallel random access machine](https://en.wikipedia.org/wiki/Parallel_random_access_machine" \o "Parallel random access machine) (PRAM) model. BSP differs from PRAM by not taking communication and synchronization for granted. An important part of analyzing a BSP algorithm rests on quantifying the synchronization and communication needed.

**ARCHITECTURE**

Apache Hama, based on Bulk Synchronous Parallel model[1], comprises three major components:

* BSPMaster
* [GroomServer](https://wiki.apache.org/hama/GroomServer)
* Zookeeper.

It is very similar with Hadoop architecture, only except the portion of communication and synchronization mechanisms.

In a normal use-case the user submits a so called "Job" which is a definition of how to run a computation. A job once submitted will have multiple tasks that are launched across the cluster.

### BSPMaster

[BSPMaster](https://wiki.apache.org/hama/BSPMaster) is responsible for the following:

1. Maintaining its own [state](https://wiki.apache.org/hama/BSPMaster).
2. Maintaining groom server status.
3. Maintaining super-steps and other counters in a cluster.
4. Maintaining [jobs and tasks](https://wiki.apache.org/hama/JobAndTask).
5. Scheduling Jobs and assigning tasks to groom servers.
6. Distributing execution classes and configuration across groom servers.
7. Providing users with the cluster control interface (web and console based).

A BSP Master and multiple grooms are started by the script. Then, the BSP master starts up with a RPC server to which groom servers can dynamically register itself. Groom servers starts up with a BSPPeer instance - later, BSPPeer needs to be integrated with GroomServer - and a RPC proxy to contact the BSP master. After started, each groom periodically sends a heartbeat message that encloses its groom server status, including maximum task capacity, unused memory, and so on.

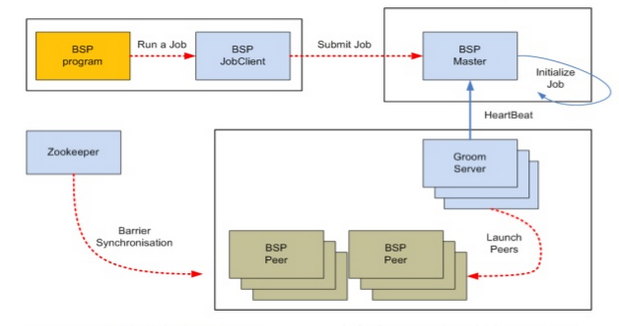
Each time the BSP master receives a heartbeat message, it brings up-to-date groom server status - the BSP master makes use of groom servers' status in order to effectively assign tasks to idle groom servers - and returns a heartbeat response that contains assigned tasks and others actions that a groom server has to do. For now, we have a FIFO job scheduler and very simple task assignment algorithms.

### GroomServer

A [Groom Server](https://wiki.apache.org/hama/GroomServer) (shortly referred to as groom) is a process that manages life cycle of BSP tasks assigned by BSPMaster. Each groom contacts the BSPMaster, and reports task statuses by means of periodical piggybacks with BSPMaster. Each groom is designed to run with HDFS or other distributed storages. Basically, a groom server and a data node should run on one physical node to get the best performance for data-locality. Note that in a massive parallel environment, the benefit of data locality is lost when large amount of virtual processes must be multiplexed onto physical processes.

### Zookeeper

A Zookeeper is used to manage the efficient barrier synchronization of the BSPPeers. Later, it will also be used for the area of a fault tolerance system.



**APPLICATION**

1. It is a [bridging model](https://en.wikipedia.org/wiki/Bridging_model" \o "Bridging model) for designing [parallel algorithms](https://en.wikipedia.org/wiki/Parallel_algorithm" \o "Parallel algorithm).
2. It is massive scientific calculations.
3. It can also be used as Distributed Computing Framework.
4. It has a very large computations that can exceeds the abilities of Map-Reduce Programming.
5. It can also used in Machine Learning. **Example**: - Neuron-centric programming model - The new Programming API proposed by Edward J. Yoon will provide two user-defined functions, which the user can define the characteristic of artificial neural network model: Activation function and Cost function. Each function can be implemented by extending Activation Function and Cost Function abstract classes.
6. It can also be used in Matrix based Algorithms.
7. It is used to perform [graph processing](https://en.wikipedia.org/wiki/Graph_(computer_science)" \o "Graph (computer science)) on [big data](https://en.wikipedia.org/wiki/Big_data" \o "Big data).
8. It can also be used in Network Algorithms.

**FEATURES**

Apache Hama will develop a parallel matrix computational package based on [Hadoop](http://hadoop.apache.org) Map/Reduce. Hama will develop a parallel matrix computational package, which provides an library of matrix operations for the large-scale processing development environment and Map/Reduce framework for the large-scale Numerical Analysis and Data Mining, which need the intensive computation power of matrix inversion, e.g. linear regression, PCA, SVM and etc. It will be also useful for many scientific applications, e.g. physics computations, linear algebra, computational fluid dynamics, statistics, graphic rendering and many more.

Currently, several shared-memory based parallel matrix solutions can provide a scalable and high performance matrix operations, but matrix resources can not be scalable in the term of complexity. And, Hadoop HDFS Files and Map/Reduce can only used by 1D blocked algorithm.Hama approach proposes the use of 3-dimensional Row and Column (Qualifier), Time space and multi-dimensional Column-families of [Hbase](http://hadoop.apache.org/hbase), which is able to store large sparse and various type of matrices (e.g. Triangular Matrix, 3D Matrix, and etc.) and utilize the 2D blocked algorithm. its auto-partitioned sparsity sub-structure will be efficiently managed and serviced by Hbase. Row and Column operations can be done in linear-time, where several algorithms, such as structured Gaussian elimination or iterative methods, run in O(the number of non-zero elements in the matrix / number of mappers) time on Hadoop Map/Reduce.

It is used to perform [graph processing](https://en.wikipedia.org/wiki/Graph_(computer_science)" \o "Graph (computer science)) on [big data](https://en.wikipedia.org/wiki/Big_data" \o "Big data). It make the use of Giraph which utilizes [Apache Hadoop](https://en.wikipedia.org/wiki/Apache_Hadoop" \o "Apache Hadoop)'s Map-Reduce implementation to process graphs. [Facebook](https://en.wikipedia.org/wiki/Facebook" \o "Facebook) used Giraph with some performance improvements to analyze one trillion edges using 200 machines in 4 minutes.

**Efficiency on graph problems.** The framework is designed to support iterative computations more efficiently than Map-Reduce, because it keeps the dataset in memory rather than writing it to disk after every iteration. This is useful since many graph algorithms are iterative. It also handles the fact that graph algorithms generally have poor memory access locality, by locating different vertices on different machines and passing messages between machines as necessary.

**Ease of programming.** The programming model is natural when working with graphs because it makes vertices and edges first-class citizens and encourages programmers to think in those terms, rather than in terms of data flows and transformation operators on parts of the graph.

**ADVANTAGES**

1. Easily programmable.
2. Highly efficient on Graph Based Problem.
3. It will be also useful for many scientific applications, e.g. physics computations, linear algebra, computational fluid dynamics, statistics, graphic rendering and many more.
4. It provides an library of matrix operations for the large-scale processing development environment and Map/Reduce framework for the large-scale Numerical Analysis and Data Mining.
5. It can also used in Machine Learning, Network Algorithm and also in designing of Parallel Algorithms.

**DISADVANTAGES**

1. In its current state, the 'Hama' is buggy and needs filling out, but generalized matrix interface and basic linear algebra operations was implemented within a large prototype system. In the future, We need new parallel algorithms based on Map/Reduce for performance of heavy decomposition and factorization. It also needs tools to compose an arbitrary matrix only with certain data filtered from Hbase array structure.
2. In Hama, communication and synchronization processes are relatively slow. Outgoing Message Manager collects the message to be sent, serializes it, compresses it and puts it in a bundles. At barrier synchronization phase, each BSP task exchanges the bundles, deserializes it, decompresses it and puts it into the Incoming Queue.

**CONCLUSION**

Apache Hama is very good tool for dealing with very large data. Its ability is even more then the data handles by the Map-Reduce Programming. It is even much more easier to program using Apache Hama. It is generally very much efficient in Graph Based Algorithms, Matrix Operations, Network Based Algorithms. It can be easily use for very large scientific computation. It can also be easily used in Machine Learning Algorithms such as Principal Component Analysis, Support Vector Machine, etc. But right now, Apache Hama is bit buggy right now, and needs to filled out.