Architecture

Components

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

- BSPMaster
- GroomServer
- Zookeeper.

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

In a normal usecase 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 is responsible for the following:

- Maintaining its own state.
- Maintaining groom server status.
- Maintaining supersteps and other counters in a cluster.
- Maintaining jobs and tasks.
- Scheduling Jobs and assigning tasks to groom servers
- Distributing execution classes and configuration across groom servers.
- 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 (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[2].

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.

Communication and Synchronization Process

Each BSP task has a set of Outgoing Message Manager and Incoming Queue.

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.

System Diagram

? Unknown Attachment

1. BSPMaster starts up
2. GroomServer starts up
3. ZooKeeper cluster starts up

4. GroomServer dynamically registers itself to BSPMaster
5. GroomServer forks/ manages BSPPeer(s)
6. BSPPeers communicate/ perform barrier synchronization through ZooKeeper cluster.

Reference

[1]. Valiant, Leslie G., A bridging model for parallel computation.

[2]. David B. Skillicorn, Jonathan M. D. Hill, and W. F. [McColl]. Questions and Answers about BSP. Scientific Programming, 6(3):249-274, Fall 1997.