Useful links (the proposal starts next page)

GSoC JIRA Issue

https://issues.apache.org/jira/browse/WHIRR-529

Create Whirr services slides

http://www.slideshare.net/jclouds/how-to-write-a-whirr-service?from=ss_embed

Hive slides

http://www.slideshare.net/namit_jain/hadoop-summit-2009-hive

Giraph presentation

http://www.youtube.com/watch?v=l4nQjAG6fac&feature=plcp&context=C48a3032VDvjVQa1PpcFMIvy-my8jRREjpv7H74dKnOsw_mp8xRSo=

Sqoop site

https://github.com/cloudera/sqoop/wiki

Kafka presentation

http://www.youtube.com/watch?v=Eq3i2m8aJBI&feature=plcp&context=C49acdaeVDvjVQa1PpcFMIvy-my8jRRCz4bQSNdjKyOf5acv4H8dA=

Apache proposal examples

http://google-melange.appspot.com/gsoc/proposal/review/google/gsoc2011/nirmal070125/1

https://issues.apache.org/jira/browse/MAHOUT-627

Whirr Proposal

Abstract

Apache Whirr is a set of libraries to provide a common interface for running cloud services (in a cloud-neutral way). It abstracts the idiosyncrasies of each provider and has a common service API. The main goal of this project is bring to Whirr the capacity to support the execution of new services: Sqoop, Hive, Giraph, Kafka and Flume. Also, as a stretch goal, the needed functionalities to have a multicloud Elastic MapReduce support. These new features should bring to Whirr a considerable improvement and also an even large range of users.

Detailed description

Apache Whirr is a set of libraries to provide a common interface for running cloud services (in a cloud-neutral way). It abstracts the idiosyncrasies of each provider, so you don't have to concern about configuration details and has a common service API, which facilitates the conception of new features.

Currently, Whirr has support for some cloud providers and services, since the most established and used to the most recent ones. As currently supported services, we can cite Hadoop, Zookeeper, Mahout, Cassandra, Puppet, and others. Such features provided by Whirr, turns it in a "polyglot" way to manage services in the cloud with a single (common) interface.

In this way, to help with Whirr's evolution and constant improvement, this proposal main goal is to bring new services that will help achieve an even larger community of users and contributors:

- Sqoop: tool designed to transfer data between Hadoop and structured datastores such as relational databases;
- Giraph: large-scale, fault-tolerant, Bulk Synchronous Parallel (BSP)-based graph processing framework;
- Flume: distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data;
- Hive: data warehouse system for Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop compatible file systems;
- Kafka: distributed publish-subscribe messaging system.

As cited before, Whirr provides an API for implementing new services, a tutorial by Thom White can be found at Whirr's homepage [1]. Below is described the process of providing a new service in resumed steps:

1. Identify service roles: is basically identify the big (important) modules of a service, for each one, a role must be created;
2. Write a `ClusterActionHandler` for each identified role;
3. Write scripts to run at cluster nodes, these script must contains configuration steps needed for running the service;
4. Package and install: each service is a self-contained *jar*;Identify service roles: is basically identify the big (important) modules of a service, for each one, a role must be created;
5. Run: use a *java* class or a properties file to describe the specific configurations.

Follow a tutorial and run simple examples would help a deeper knowledge about each service structure before start the development itself.

Another great feature that should be discussed as a stretch goal would be the support for multicloud Elastic MapReduce. Its primary intention is enable a way to process vast amounts of data using the Hadoop framework. It should have a very “rock solid” behavior, what brings the need to define some requisites:

- Hadoop deployment must be robust;
- CLI for submitting and monitoring jobs;
- DistCp [2] from blobstores (big storage providers) to Hadoop/Hive cluster.

A good explanation about the Elastic MapReduce concept can be found at AWS EMR documentation [3].

Roadmap

From the following schedule, what will be fully showable at mid-term evaluation will be the full support for two or three services, depending on how the development will take place. It is important to notice that each of the proposed features have to be properly tested (automatic and real execution) and documented.

Below are the steps I am planning to take during this project:

Community Bonding Period (April 24 - May 20)

During the Community Bonding Period, the main intention will be to expand the knowledge at Whirr/Jclouds code and architecture by writing small bug fixes and patches. Also, it will serve to be more familiar with the new features details, like the services internals.

Project start until the mid-term evaluation (May 21 - July 9)

Begin the development of new services: Sqoop, Giraph and Flume. They must be fully

implemented (following the steps described before), tested and documented.

Mid-term until the project end (July 10 - August 13)

- **July 10 - August 6:** Implement the support for the remaining services: Hive and Kafka;
- **August 7 - August 13:** Final touch, with the execution of more tests and documentation, this period will be useful to finish all possible remaining details of the project.

The order that the services will be implemented can change to cover any possible not considered requisite: relevancy for the community, difficulties during their conception or even because some version update.

Stretch Goals and Future Work

In case of the proposed features development occurs faster than expected, as a good stretch goal the multicloud Elastic MapReduce support development can be discussed. It is a complex task, and some community collaboration would be very helpful during its conception.

I intend to become a Whirr contributor (maybe acquiring the status of committer :D). As I use it during my MSc research, would be great to help its evolution.

Progress Reports

The project progress, new ideas and problems will be reported regularly at my personal blog [4].

Currently, I am familiar with most of the proposal parts, such as the major structure of Whirr, like its *core* and the API to build new services (have read all the documentation available and various separate tutorials as well).

About me

I'm currently at beginning of my Master's degree at Federal University of Campina Grande, under supervision of Dr. Andrey Brito. My research is in the area of scalable distributed systems and Smart Grids, mainly at continuous (stream) processing of large amount of data.

Since May 2008 I have started my open source development experiences (https://www.ohloh.net/accounts/rodrigods) as part of OurGrid's (http://www.ourgrid.org/) team -

an open source middleware for P2P grid computing, and was responsible for some of its new releases. Its code can be found at: http://svn.lsd.ufcg.edu.br/repos/ourgrid (user *anonymous* and no password).

I have also participated of some programming competitions and achieved Honor Mention at ACM Brazilian First Phase Programming Contest [5] in 2009 and Gold Medal at “Olimpíada Paraibana de Informática - OPI” [6] in 2011.

References

[1] https://cwiki.apache.org/confluence/display/WHIRR/Implementing+a+New+Service

[2] http://aws.amazon.com/documentation/elasticmapreduce/

[3] http://hadoop.apache.org/common/docs/current/distcp.html

[4] http://rodrigodsousa.blogspot.com

[5] http://cm.baylor.edu/welcome.icpc

[6] http://www.dsc.ufcg.edu.br/~opi/