Distributed Computing using CloudLab

Linh B. Ngo
Clemson University
Introduction

• Distributed and Cluster Computing (CPSC 3620)
• Offered twice per academic year
• Average class size: 40-45 students
• Required junior-level class (typically taken at senior year)
• Contents:
  – Infrastructure/System-oriented
  – Performance/Efficiency
  – High Performance Computing
    • MPI
  – Big Data Computing
    • Hadoop MapReduce
    • Apache Spark
    • HPCCSystems
Computing Resources

• Palmetto Supercomputer
  – 2000+ nodes, open to all faculty/students
  – No administrative access
  – Cannot share nodes among students to support group assignment
  – Preemption from node owners

• CloudLab
  – Limited resources for large-scale study
  – Administrative access
  – Ease of collaboration
  – No preemption
Computing Resources

• Combine both local computing resources and CloudLab

• Learning outcomes through CloudLab
  – Administrative skills for distributed systems
  – In-depth understanding of distributed systems

• Learning outcomes through Palmetto
  – Basic understanding of parallel application development
  – Impacts of scaling and efficiency on larger systems
Tutorial

• Set up environments for distributed computing on CloudLab

• MPI
  – Two-node cluster
  – OpenMPI

• Hadoop
  – Three-node cluster
  – Hortonwork Distribution
CloudLab - Instantiate a Profile

The Apit cluster is temporarily unavailable; not likely to be up before the end of the week.

Current Usage: 0 Node Hours, Prev Week: 36, Prev Month: 39 (30 day rank: 136 of 176 users)

Profile: OpenStack

Provides a highly-configurable OpenStack instance with a controller, network manager, and one or more compute nodes (potentially at multiple CloudLab sites). This profile runs x86 or ARM64 nodes. It sets up OpenStack Juno (on Ubuntu 15.10, 15.04, or 14.10) according to your choice, and configures all OpenStack in some VM disk images, and creates basic networks accessible via floating IPs. You'll be able to create access them over the Internet in just a few minutes. When you click the Instantiate button, you'll be a list of parameters that you can change to control what your OpenStack instance will look like; the parameter documentation on that page (or in the Instructions) to understand the various features available to you.

Copy Profile Show Profile

Change Profile
The APT cluster is down due to multiple disk failures; not likely to be up before the end of the week.

Current Usage: 0 Node Hours, Prev Week: 39, Prev Month: 39 (30 day rank: 136 of 176 users)
Both nodes should have the same configuration, no network connection is needed (due to public IP)
• You can save multiple versions of your topology
• Instantiate the version that you want to launch
• The launching procedure will be similar to the OpenStack tutorial
MPI on CloudLab

On each node:

```
sudo apt-get update
sudo apt-get install libibnetdisc-dev

sudo nano /etc/environment

PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
/sbin:/bin:/usr/games:/usr/local/games:/home/mpi
user/.openmpi/bin"

LD_LIBRARY_PATH="/lib:/usr/lib:/usr/local/lib:/home/mpi
user/.openmpi/lib/"

sudo adduser mpiuser```
MPI on CloudLab

```bash
sudo adduser mpiuser
ssh mpiuser@localhost

wget https://www.open-mpi.org/software/ompi/v1.8/downloads/openmpi-1.8.1.tar.gz

tar xzf openmpi-1.8.1.tar.gz
cd openmpi-1.8.1

./configure --prefix="/home/mpiuser/.openmpi"
makes
make install
```
MPI on CloudLab

```bash
ssh-keygen -t rsa

cd .ssh

cp id_rsa.pub authorized_keys

ssh-copy-id -i id_rsa.pub <hostname of the other node>

nano nodelist

<hostname of first node>
<hostname of second node>
...
```
Example program

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/utsname.h>
#include <mpi.h>

int main(int argc, char *argv[]){
    int rank, size;
    MPI_Status status;
    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);

    struct utsname uts;
    uname(&uts);
    printf("%d at %s\n",rank,uts.nodename);
    MPI_Finalize();
    return 0;
}
```
MPI on CloudLab

mpicc gethostname.c -o gethostname

scp gethostname mpiuser@<the other node>:~/home/mpiuser
scp nodelist mpiuser@<the other node>:~/home/mpiuser

mpirun -np 2 -machinefile nodelist ./gethostname

mpirun -np 2 -machinefile nodelist --map-by node ./gethostname
Assignment Ideas

• Develop a work queue using various allocation strategies:
  – Normal
  – Cyclic
  – Dynamic

• Setup MPI cluster with nodes on separate sites, reduce network connection, and evaluate performance on different allocation strategies
Hadoop on CloudLab

- Enterprise Hadoop
- Hortonworks
Hadoop on CloudLab

- Both nodes should have the same configuration on bare metal PC, no network connection is needed (due to public IP)
On each node

- SSH onto the node from Palmetto
- Change to root:
  ```
sudo su -
  ```
- Execute the following commands:
  ```
chkconfig --list ntpd
chkconfig ntpd on
service ntpd start
chkconfig iptables off
/etc/init.d/iptables stop
setenforce 0
  ```
On each node

• Setup Ambari download server
  
  wget -nv http://public-repo-1.hortonworks.com/ambari/centos6/2.x/updates/2.1.2/ambari.repo -O /etc/yum.repos.d/ambari.repo

• On namenode
  
  yum -y install ambari-server
  
  yum -y install ambari-agent

• On datanode
  
  yum -y install ambari-agent
On namenode

• Set up ambari server:
  ambari-server setup

• Select default for all questions

• Select 1 for JDK version

• When all done, start ambari server
  ambari-server start
On each node

- Using vim to edit `/etc/ambari-agent/conf/ambari-agent.ini`
- Change:
  
  ```
  hostname=<hostname of namenode as shown in list view of CloudLab>
  ```

- Start Ambari Agent
  
  ```
  ambari-agent start
  ```
Ambari Server (admin/admin)
Ambari Server (admin/admin)

Get Started

This wizard will walk you through the cluster installation process. First, start by naming your new cluster.

Name your cluster Learn more
Test

Next →
Ambari Server (admin/admin)

Select Stack

Please select the service stack that you want to use to install your Hadoop cluster.

Stacks

- HDP 2.3
- HDP 2.2
- HDP 2.1
- HDP 2.0

Advanced Repository Options

← Back Next →
Install Options

Enter the list of hosts to be included in the cluster and provide your SSH key.

Target Hosts

Enter a list of hosts using the Fully Qualified Domain Name (FQDN), one per line. Or use Pattern Expressions

cinode095.clemson.cloudlab.us
cinode092.clemson.cloudlab.us

Host Registration Information

- Provide your SSH Private Key to automatically register hosts

Choose File: No file chosen

ssh private key

- Perform manual registration on hosts and do not use SSH

Register and Confirm →
Assuming you had ambari agents up and running ...
Ambari Server (admin/admin)

- HDFS
- YARN+MapReduce2
- Tez
- ZooKeeper
- Ambari Metrics
Assign Masters

Assign master components to hosts you want to run them on.

SNameNode: clnode095.clemson.cloudlab.us (): ▼
NameNode: clnode095.clemson.cloudlab.us (): ▼
History Server: clnode095.clemson.cloudlab.us (): ▼
App Timeline Server: clnode095.clemson.cloudlab.us (): ▼
ResourceManager: clnode095.clemson.cloudlab.us (): ▼
ZooKeeper Server: clnode095.clemson.cloudlab.us (): ▼
ZooKeeper Server: clnode092.clemson.cloudlab.us (252.2 GB, 20 cores)
Metrics Collector: clnode095.clemson.cloudlab.us (): ▼

Save and Next ☰
### Assign Slaves and Clients

Assign slave and client components to hosts you want to run them on. Hosts that are assigned master components are shown with *.

*"Client" will install HDFS Client, MapReduce2 Client, YARN Client, Tez Client and ZooKeeper Client.*

<table>
<thead>
<tr>
<th>Host</th>
<th>all</th>
<th>none</th>
<th>all</th>
<th>none</th>
<th>all</th>
<th>none</th>
<th>all</th>
<th>none</th>
</tr>
</thead>
<tbody>
<tr>
<td>cinode095.clemson.cloudlab.us</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cinode095.clemson.cloudlab.us</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cinode092.clemson.cloudlab.us</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Show: 25 1-2 of 2 | Next →
Edit configuration as you see fit

**NameNode**
- **NameNode directories**
  - /hadoop/hdfs/namenode
- **NameNode Java heap size**
  - 128 GB
- **NameNode Server threads**
  - 256
- **Minimum replicated blocks %**
  - 100%

**DataNode**
- **DataNode directories**
  - /hadoop/hdfs/data
- **DataNode failed disk tolerance**
  - 0
- **DataNode maximum Java heap size**
  - 128.7GB
- **DataNode max data transfer threads**
  - 4096
Deploy ...
Warning due to lack of space and failed checks (ignore)
Overview 'cn0node095.clemson.cloudlab.us:8020' (active)

Version: 2.7.1.2.3.2.0-2950, r5cc60e0600e33aa96205f18bcca9af36cb193c1c
Compiled: 2015-09-30T18:08Z by jenkins from HEAD detached at 5cc60e0
Cluster ID: CID-c31ab4be-d8d4-42a9-b456-7745974a5b1d
Block Pool ID: BP-160930222-130.127.133.104-1444275269044

Summary

Security is off.
Safe-mode is off.
38 files and directories, 11 blocks = 49 total filesystem object(s).
Heap Memory used 15.41 GB of 123.44 GB Heap Memory. Max Heap Memory Is 123.44 GB.
Non Heap Memory used 50.15 MB of 61.21 MB Committed Non Heap Memory. Max Non Heap Memory Is -1 B.

Configured Capacity: 14.75 GB
DFS Used: 0.00 MB (0.00%)
## YARN

The image shows a screenshot of a Hadoop YARN cluster management interface. The page displays the 'All Applications' section of the Hadoop YARN system. The table contains detailed information about various applications running on the cluster, including:

- **ID** of the application
- **User** who submitted the application
- **Name** of the application
- **Application Type** (YARN, TEZ)
- **Queue** in which the application is running
- **StartTime** and **FinishTime** of the application
- **State** of the application (FINISHED, FAILED)

The table entries illustrate applications running on the cluster, with statuses indicating successful completion or failure. The interface also shows the cluster metrics and scheduler metrics, providing insights into the resource utilization and scheduling process within the Hadoop YARN environment.
sudo su hdfs
hdfs dfs -mkdir /user/<username>
hdfs dfs -chown <username>:<username> /user/<username>
exit to <username>
hdfs dfs -ls /user/
git clone https://github.com/clemsoncoe/Introduction-to-Hadoop-data.git
cd Introduction-to-Hadoop-data
hdfs dfs -put gutenberg-shakespeare.txt /user/<username>/
yarn jar /usr/hdp/current/hadoop-mapreduce-client/hadoop-mapreduce-examples-2.7.1.2.3.6.0-3796.jar wordcount
gutenberg-Shakespeare.txt output/
hdfs dfs -ls output
hdfs dfs -cat output/part-r-00000
Assignment Ideas

• Deploy a Hadoop cluster and upload a large data set (Airline on-time performance data: http://stat-computing.org/dataexpo/2009/the-data.html)

• Examine and investigate performance of Hadoop MapReduce as data nodes are killed/added to the cluster

• Examine performance of Hadoop MapReduce as data nodes are located on different sites