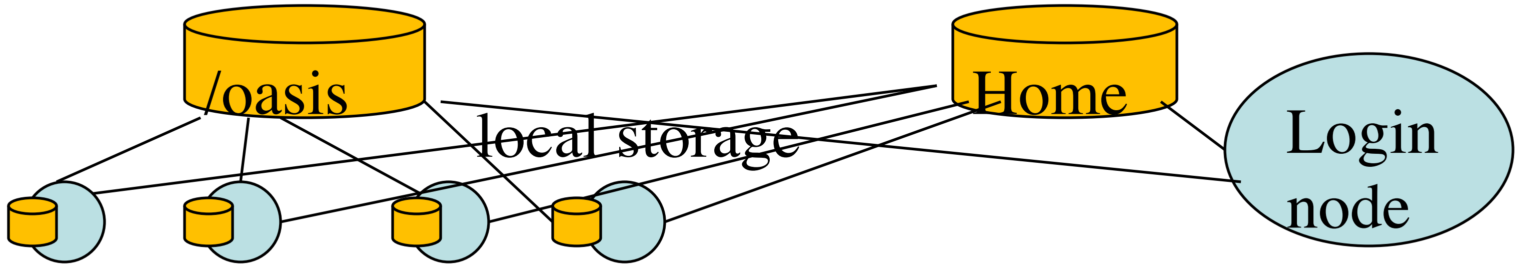
Spark Programming at Comet

UCSB CS240A 2017. Tao Yang

Comet Cluster

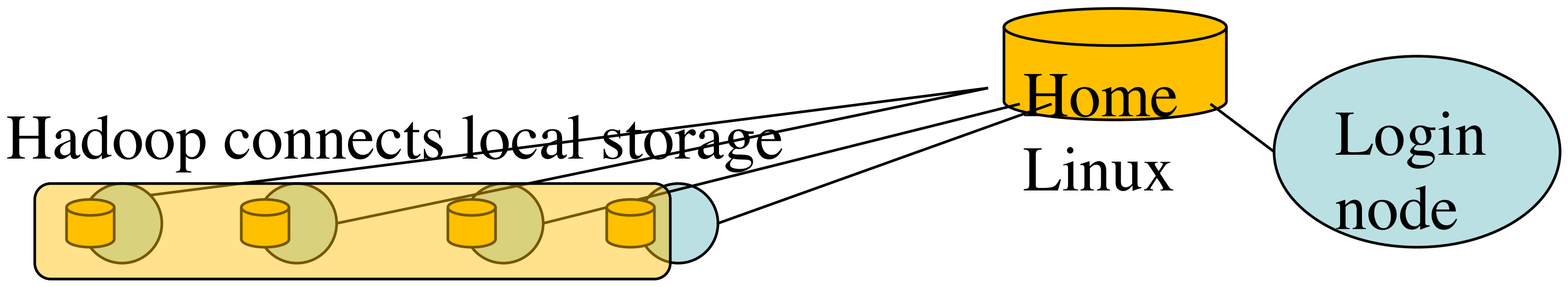
- Comet cluster has 1944 nodes and each node has 24 cores, built on two 12-core Intel Xeon E5-2680v3 2.5 GHz processors
- 128 GB memory and 320GB SSD for local scratch space.
- Attached storage: Shared 7 petabytes of 200 GB/second performance storage and 6 petabytes of 100 GB/second durable storage
  - Lustre Storage Area is a Parallel File System (PFS) called Data Oasis.
    - Users can access from

/oasis/scratch/comet/$USER/temp_project



Hadoop installation at Comet

- ***Installed in /opt/hadoop/1.2.1***
- **Configure Hadoop on-demand with myHadoop:**
  - /opt/hadoop/contrib/myHadoop/bin/myhadoop-configure.sh



Hadoop file system is built dynamically on the nodes allocated. Deleted when the allocation is terminated.

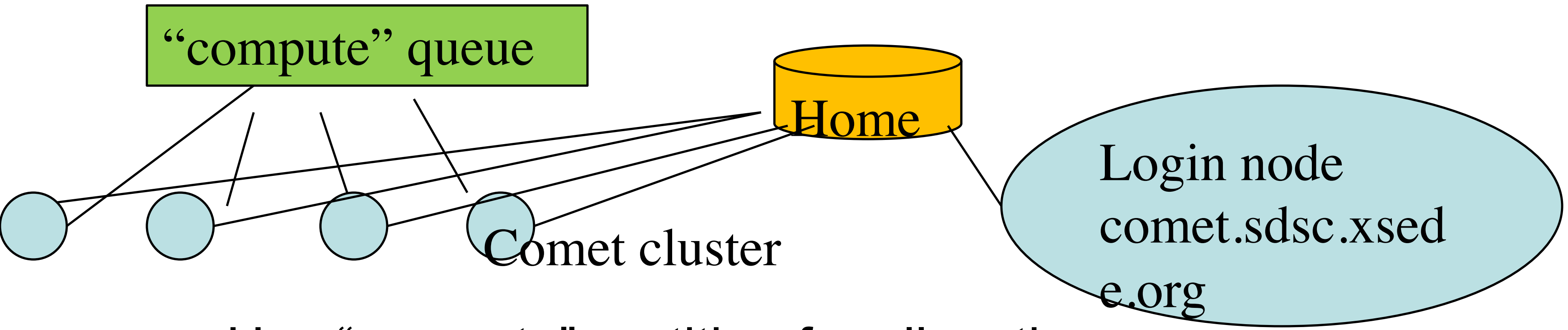
Sample Spark Programs

Sample from Comet team is in /home/tyang/cs240sample/sparkgraphx.

Spark word count example is available at Comet under /home/tyang/cs240sample/spark-wc/

- **wordcount.py** – Python wordcount code using Spark.
- **docwordcount.py** for counting the number of documents each word appears.
- **Makefile** - instructions to submit and run a Spark python job
- **Importer.java** – Convert the input format from .txt to .seq sequence format using Hadoop library. It splits the input file into separate (key, value) pairs. The key is arbitrary (like "doc_xyz") and the value will be the contents of our input file between two "---END.OF.DOCUMENT---" markers.
  - Use “sbatch submit-hadoop-importer.sh” to submit and run.

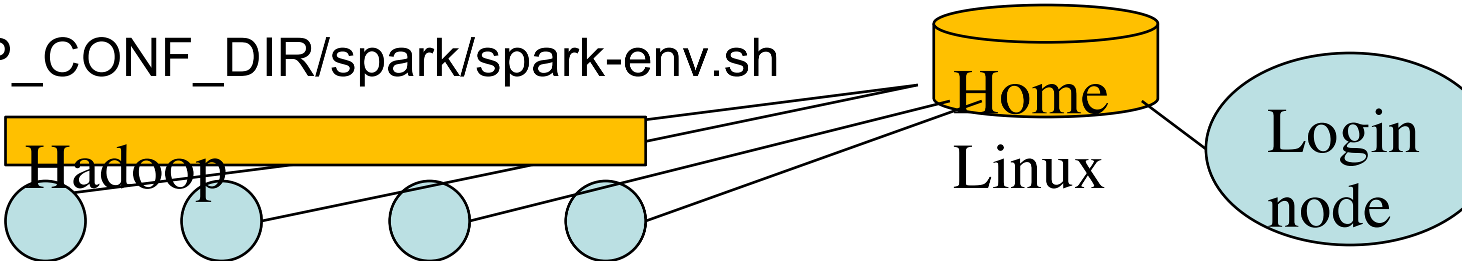
How to Run a Spark Job



- Use "compute" partition for allocation
- sbatch submit-spark-wc.sh
  - Data input is data/billOfRights1.txt.seq
  - Data output is in spark-wc-out
- Job trace sample is sparkwc.1570908.comet-18-08.out

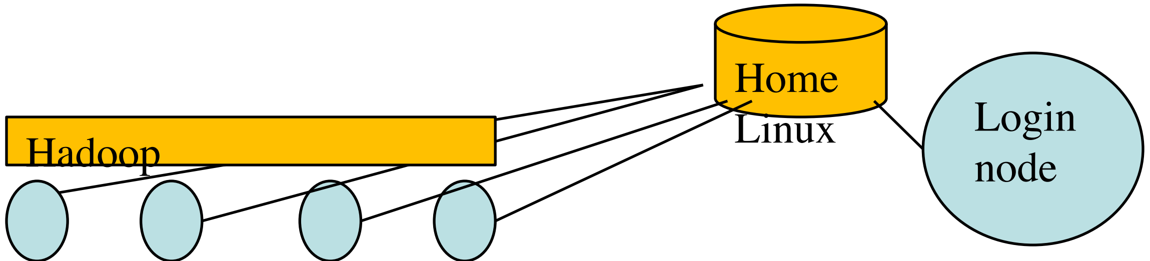
Sample script (submit-spark-wc.sh)

```
#!/bin/bash
#SBATCH --job-name="sparkpython-demo"
#SBATCH --output="sparkwc.%j.%N.out"
#SBATCH --partition=compute
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=2
#SBATCH --export=ALL
#SBATCH -t 00:30:00 module load spark
export PATH=/opt/hadoop/2.6.0/sbin:$PATH
export HADOOP_CONF_DIR=$HOME/mycluster.conf
export WORKDIR=`pwd`
#Build a Hadoop file system
myhadoop-configure.sh
#Start all demons of Hadoop/Spark.
start-dfs.sh
source $HADOOP_CONF_DIR/spark/spark-env.sh
myspark start
```



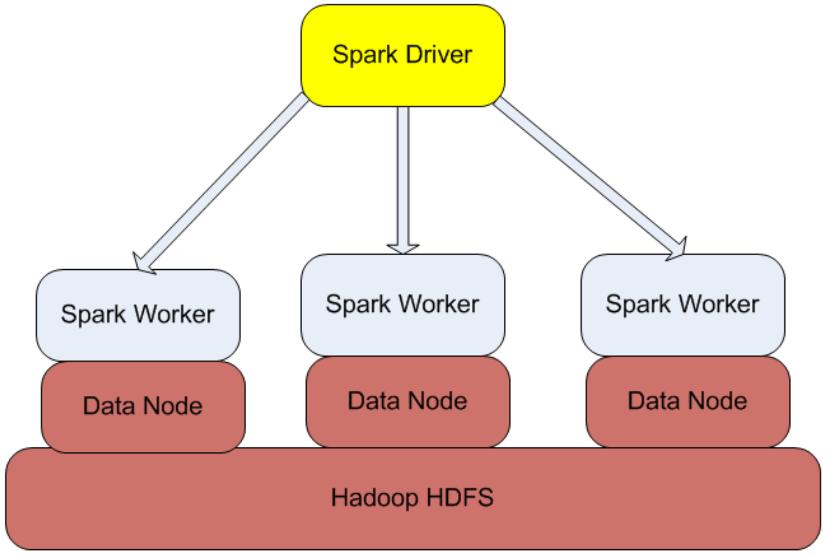
Sample script

```
#make an input directory in the hadoop file system
hdfs dfs -mkdir -p /user/$USER
#copy data from local Linux file system to the Hadoop file system
hdfs dfs -put $WORKDIR/data/billOfRights1.txt.seq
    /user/$USER/input.seq
#Run Spark Python wordcount job
spark-submit wordcount.py /user/$USER/input.seq output
# Create a local directory  to host the output data
rm -rf spark-wc-out >/dev/null || true
mkdir -p spark-wc-out
# Copy out the output data
hadoop dfs -copyToLocal output/part* spark-wc-out
#Stop all demons and cleanup
myspark stop
stop-dfs.sh
myhadoop-cleanup.sh
```



Sample output trace wordcount.1569018.comet-17-14.out

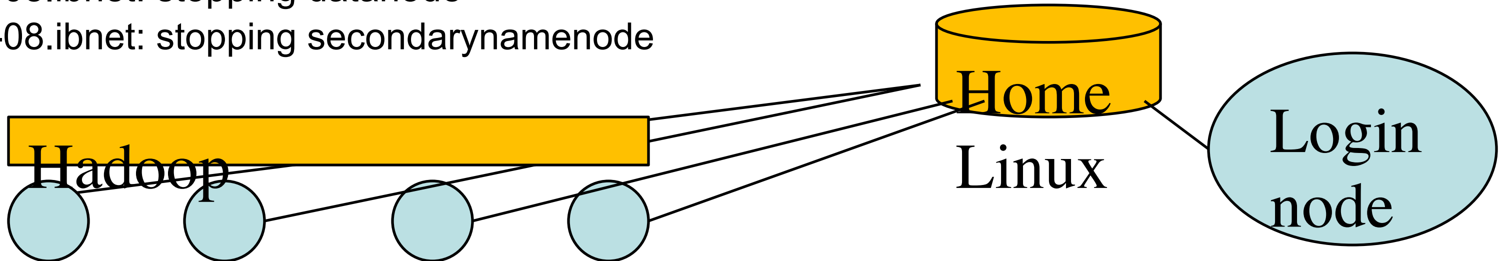
```
comet-18-08.ibnet: starting namenode, logging to /scratch/tyang/1570908/logs/hadoop-tyang-namenode-
      comet-18-08.sdsc.edu.out
comet-18-08.ibnet: starting datanode, logging to /scratch/tyang/1570908/logs/hadoop-tyang-datanode-
      comet-18-08.sdsc.edu.out
comet-18-09.ibnet: starting datanode, logging to /scratch/tyang/1570908/logs/hadoop-tyang-datanode-
      comet-18-09.sdsc.edu.out
comet-18-08.ibnet: starting secondarynamenode, logging to /scratch/tyang/1570908/logs/hadoop-tyang-
      secondarynamenode-comet-18-08.sdsc.edu.out
starting org.apache.spark.deploy.master.Master, logging to /scratch/tyang/1570908/logs/spark-tyang-
      org.apache.spark.deploy.master.Master-1-comet-18-08.out
starting org.apache.spark.deploy.worker.Worker, logging to /scratch/tyang/1570908/logs/spark-tyang-
      org.apache.spark.deploy.worker.Worker-1-comet-18-08.sdsc.edu.out
starting org.apache.spark.deploy.worker.Worker, logging to /scratch/tyang/1570908/logs/spark-tyang-
      org.apache.spark.deploy.worker.Worker-1-comet-18-09.sdsc.edu.out
```



Sample output trace wordcount.1569018.comet-17-14.out

```
16/02/01 11:30:08 INFO executor.Executor: Finished task 0.0 in stage 0.0 (TID 0). 2437 bytes result sent
        to driver
16/02/01 11:30:08 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 0.0 (TID 0) in 161 ms on
        localhost (1/1)
16/02/01 11:30:08 INFO executor.Executor: Finished task 0.0 in stage 1.0 (TID 1). 2317 bytes result sent
        to driver
16/02/01 11:30:08 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 1.0 (TID 1) in 163 ms on
        localhost (1/1)
16/02/01 11:30:09 INFO executor.Executor: Finished task 0.0 in stage 2.0 (TID 2). 1229 bytes result sent
        to driver
16/02/01 11:30:09 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 2.0 (TID 2) in 194 ms on
        localhost (1/1)

stopping org.apache.spark.deploy.master.Master
stopping org.apache.spark.deploy.worker.Worker
stopping org.apache.spark.deploy.worker.Worker
comet-18-08.ibnet: stopping namenode
comet-18-09.ibnet: stopping datanode
comet-18-08.ibnet: stopping datanode
comet-18-08.ibnet: stopping secondarynamenode
```



Sample input and output

```
$ more data/billOfRights.txt
Amendment I
Congress shall make no law respecting an establishment of religion, or prohibiting the free
      exercise thereof; or abridging
the freedom of speech, or of the press; or the right of the people peaceably to assemble, and
      to petition the Government for a redress of grievances.
---END.OF.DOCUMENT---
Amendment II
A well regulated Militia, being necessary to the security of a free State, the right of the people
      to keep and bear Arms, s
hall not be infringed.
---END.OF.DOCUMENT---

$  more spark-wc-out/part-00000
(u'all', 1)
(u'United', 2)
(u'particularly', 1)
(u'just', 1)
(u'being', 1)
(u'consent', 1)
(u'supported', 1)
(u'Suits', 1)
(u'press', 1)
(u'same', 1)
(u'committed', 1)
```

Shell Commands for Hadoop File System

- **Mkdir, ls, cat, cp**
  - hadoop dfs -mkdir /user/deepak/dir1
  - hadoop dfs -ls /user/deepak
  - hadoop dfs -cat /usr/deepak/file.txt
  - hadoop dfs -cp /user/deepak/dir1/abc.txt /user/deepak/dir2
- **Copy data from the local file system to HDF**
  - hadoop dfs -copyFromLocal <src:localFileSystem> <dest:Hdfs>
  - Ex: hadoop dfs –copyFromLocal /home/hduser/def.txt /user/deepak/dir1
- **Copy data from HDF to local**
  - hadoop dfs -copyToLocal <src:Hdfs> <dest:localFileSystem>

Notes

- **To check the status of your job**

  squeue -u username
- **To cancel a submitted job**

  scancel job-id
- You have to request *all* 24 cores on the nodes. Hadoop is java based and any memory limits start causing problems. Also, in the compute partition you are charged for the whole node anyway.

Notes

- Your script should delete the outout directory if you want to rerun and copy out data   to that directory. Otherwise the Hadoop copy back fails because the file already exists.

  The current script forces to remove “spark-wc-out".
- If you are running several jobs simultaneously, please make sure you choose different locations for for the configuration files. Basically change the line:

```
export HADOOP_CONF_DIR=/home/$USER/cometcluster
```

to point to different directories for each run. Otherwise the configuration from different jobs will overwrite in the same directory and cause problems.