Report: Experiment on Cloud Computing

Victrid
dept. of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai, China
github.com/Victrid

I. Introduction

As computer technology continues to cross into various fields, the resulting huge amount of data makes it challenging to analyze and process the data. Traditional single computer programs have difficulty processing these data, while mainframe clusters are expensive and require complex operations and controls, making them limited to large analytics companies.

Today, cloud computing technology is reaching into every corner of the world. As the superiority of controlled cost, elastic management, and rapid migration is gradually being demonstrated, even ordinary people are able to access these resources and take advantage of them. The development of distributed technology has also made it possible to process huge amounts of data on inexpensive computer arrays. Today's cloud computing centers have become an important part of the Internet's infrastructure and will occupy an even more important position in the future.

This report on Cloud Computing Experiment contains these parts:

- Create VM cluster and configure Hadoop and Spark framework on Huawei Cloud.
- Run The Examples from Hadoop and Spark.
- Solve actual problems with GraphX API with Spark.

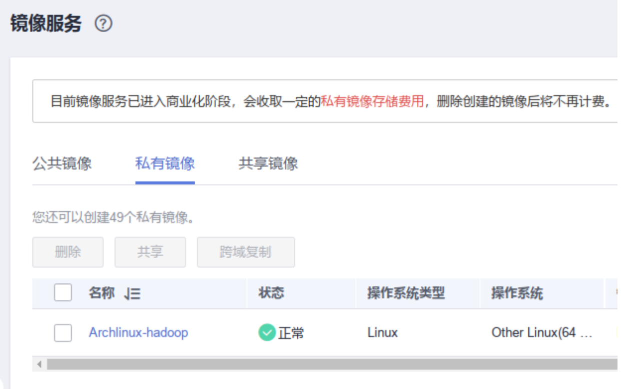
II. Configuration of Virtual Machine Cluster

A. Mirror Configuration

Fast provisioning and delivery, automation, and ease of scaling are key to today's elastic computing. However, setting up instances one by one, configuring their cumbersome dependencies and environment variables, and installing them by downloading source code and binary packages from the web and configuring them by copy and paste is typical of UNIX mainframe administrators in the 1980s.

Compared to the handbook [1] suggests that using an Ubuntu image and configure them manually, we've built a custom mirror, which already has Hadoop, Spark, and SBT packed up and modified. We referenced the image creation tool provided by Arch Linux on [2] with the Cloud-init tool and rewrote the compiling script of the Hadoop package on [3] to match the `JAVA_HOME` lookup process. A system installation script was written to suit the needs of this experiment. This series of configuration files are placed in the submitted files' `image-buildscript` folder.

Many cloud providers has provided custom image service, so do Huawei Cloud. By utilizing its IMS services [4] as figure 1, we can now create virtual machines with our pre-built images.



目前镜像服务已进入商业化阶段，会收取一定的私有镜像存储费用，删除创建的镜像后将不再计费。

您还可以创建49个私有镜像。

□	名称	状态	操作系统类型	操作系统
□	Archlinux-hadoop	✓ 正常	Linux	Other Linux(64 …

Fig. 1: Huawei Cloud IMS

B. Virtual Machine Configuration

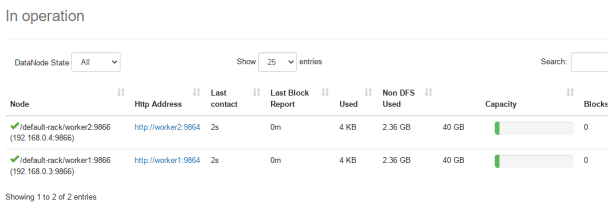
To perform VM configuration as designed in our mirror, adjusting roles while the VM is created, we use Huawei Cloud SDK and its OpenAPI to perform creation. The core part is to utilize OpenStack `user_data`. When the `user_data` is provided as bash scripts, it will be run at VM creation by cloud-init. The script we created performs Hadoop configuration according to Hadoop Documentation [5], worker appointment, and SSH key configuration. This series of configuration scripts and API call Python scripts are placed in the submitted files' `HuaweiCloud-openAPI` folder.

C. Hadoop Initiation

By running the commands below: (Our Hadoop instance is installed under `/usr/lib`)

```
cd /usr/lib/hadoop
bin/hdfs namenode -format
sbin/start-all.sh
```

The Hadoop instance will be set up, and can be viewed via `http://master:9870` as in figure 2.



In operation

DataNode State All Show 25 entries Search:

Node	Http Address	Last contact	Last Block Report	Used	Non DFS Used	Capacity	Blocks
✓/default-rack/worker2:9866 (192.168.0.4:9866)	http://worker2:9864	2s	0m	4 KB	2.36 GB	40 GB	0
✓/default-rack/worker1:9866 (192.168.0.3:9866)	http://worker1:9864	2s	0m	4 KB	2.36 GB	40 GB	0

Showing 1 to 2 of 2 entries

Fig. 2: DataNode Information on NameNode

III. Hadoop Example: wordcount

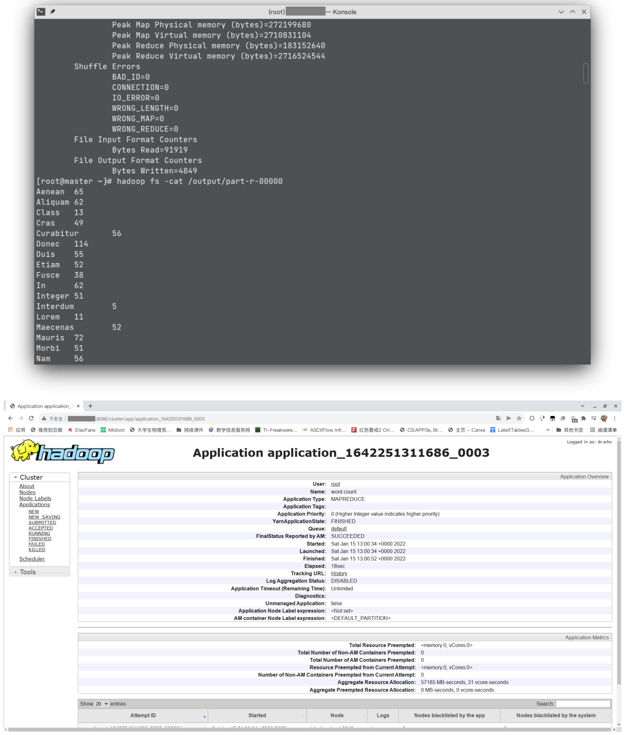
We've prepared a dummy text file, containing *lorem ipsum*, an industry standard dummy text in the printing and typesetting since 1500s. The text file contains 150 paragraphs, 13547 words, and is used to test.

The text file is transmitted to the master node via scp at /root/lorem.txt.

by running the commands below (at /root folder):

```
hadoop fs -mkdir /input
hadoop fs -put lorem.txt /input
hadoop jar /usr/lib/hadoop/share/hadoop/\
mapreduce/\
hadoop-mapreduce-examples-3.3.1.jar \
wordcount /input /output
hadoop fs -cat /output/part-r-00000
```

The running results are shown in figure 3. Both input file and results are put in the WordCount folder.



(root) [redacted] — Konsole

```
                Peak Map Physical memory (bytes)=272199680
                Peak Map Virtual memory (bytes)=2710831104
                Peak Reduce Physical memory (bytes)=183152640
                Peak Reduce Virtual memory (bytes)=2716524544
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=91919
        File Output Format Counters
                Bytes Written=4849
[root@master ~]# hadoop fs -cat /output/part-r-00000
Aenean  65
Aliquam 62
Class   13
Cras    49
Curabitur       56
Donec   114
Duis    55
Etiam   52
Fusce   38
In      62
Integer 51
Interdum        5
Lorem   11
Maecenas        52
Mauris  72
Morbi   51
Nam     56
```

Application application_1 ×

不安全 [redacted]:8088/cluster/app/application_1642251311686_0003

Logged in as: dr.who

Application application_1642251311686_0003

Application Overview

User: root
Name: word count
Application Type: MAPREDUCE
Application Tags:
Application Priority: 0 (Higher Integer value indicates higher priority)
YarnApplicationState: FINISHED
Queue: default
FinalStatus Reported by AM: SUCCEEDED
Started: Sat Jan 15 13:00:34 +0000 2022
Launched: Sat Jan 15 13:00:34 +0000 2022
Finished: Sat Jan 15 13:00:52 +0000 2022
Elapsed: 18sec
Tracking URL: History
Log Aggregation Status: DISABLED
Application Timeout (Remaining Time): Unlimited
Diagnostics:
Unmanaged Application: false
Application Node Label expression: <Not set>
AM container Node Label expression: <DEFAULT_PARTITION>

Application Metrics

Total Resource Preempted: <memory:0, vCores:0>
Total Number of Non-AM Containers Preempted: 0
Total Number of AM Containers Preempted: 0
Resource Preempted from Current Attempt: <memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt: 0
Aggregate Resource Allocation: 57185 MB-seconds, 31 vcore-seconds
Aggregate Preempted Resource Allocation: 0 MB-seconds, 0 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Nodes blacklisted by the app	Nodes blacklisted by the system

Fig. 3: Word Count Results

By running the commands below, we can clear the input files and results. This will be helpful for further experiments.

```
hadoop fs -rm -f -r /output
hadoop fs -rm -f -r /input
```

IV. GraphicX Example: Connected Component

We've copied the needed file and organized as [6]. The package needs more dependencies than WordCount program, and the scala used by Spark 3.2.0 should be 2.12.15. We modified the simple build tools script as below:

```
#    [Trailed for PDF typesetting]
scalaVersion := "2.12.15"

libraryDependencies+="org.apache.spark" \
    %% "spark-core" % "3.2.0"
libraryDependencies+="org.apache.spark" \
    %% "spark-sql" % "3.2.0"
libraryDependencies+="org.apache.spark" \
    %% "spark-graphx" % "3.2.0"
```

After compiling as in figure 4b, we upload these 2 graph txt files to data/graphx folder on HDFS, and call the class by org.apache.spark.examples.graphx.ConnectedComponentsExample as in figure 4c and received the result as in figure 4d:

```
(justinbieber,1)
(matei_zaharia,3)
(ladygaga,1)
(BarackObama,1)
(jeresig,3)
(odersky,3)
```

The necessary files and source code are put in the GraphX folder.

V. PageRank Algorithm

Here we use the Wikipedia Vote Network dataset [7] as our processing source. The requested PageRank algorithm is similar to the *Connected Component* one, and after checking the spark source code, we're sure that the GraphLoader can also be used to process the wiki votes.

PageRank algorithm is named after both the term ”web page” and Google co-founder Larry Page. The key to the ranking is a probabilistic balance between nodes. At the beginning of the computational process the pagerank for each node is randomized, and for each iteration, the rank is computed as:

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)} \tag{1}$$

An approximation of real page rank value can be calculated with several iterations.

After researching on GraphX Programming Guide [8], the graph build with GraphLoader has implemented PageRank algorithm. In our implementation, it is called by

```
val ranksGraph = graph.pageRank(0.0001)
```

The original source file does not contain name information, but we want to form a more intuitive result as *Connected Component*, which adds an additional user–node linkage. We used `sed` scripts to pre-process the original vote data from [9], and generate the user–node list as in `users.txt`.

```
sed '/^[\s]*#/d;
/^\s*$/d;
/^[ET].*/d;
/^N\t-1\tUNKNOWN.*/d;
s/^[UN]\t\(.*\)\t\(.*\)$/\1,\2/g;
s/^V\t.*\t\(.*\)\t.*\t\(.*\)$/\1,\2/g' \
original.txt | sort -g | uniq > users.txt
```

The user–node list generated would be like:

```
3,ludraman
4,gzornenplatz
5,orthogonal
6,andrevan
7,texture
8,lst27
9,mirv
...
```

After compiling and submitting like we've done in the *Connected Component* part, the results are shown below and as figure 4e:

```
PR: 32.78, ID:4037, Name:elonka
PR: 26.18, ID:  15, Name:danny
PR: 25.52, ID:6634, Name:tenpoundhammer
PR: 23.36, ID:2625, Name:_clown_will_eat_me
PR: 18.56, ID:2398, Name:werdna
PR: 17.96, ID:2470, Name:alex_bakharev
PR: 17.76, ID:2237, Name:khoikhoi
PR: 16.14, ID:4191, Name:ryulong
PR: 15.44, ID:7553, Name:dihydrogen_monoxide
PR: 15.30, ID:5254, Name:gracenotes
PR: 14.51, ID:2328, Name:phaedriel
PR: 14.48, ID:1186, Name:william_m._connolley
PR: 13.84, ID:1297, Name:robchurch
PR: 13.78, ID:4335, Name:mer-c
PR: 13.75, ID:7620, Name:cobi
PR: 13.65, ID:5412, Name:protectionbot
PR: 13.57, ID:7632, Name:redirectcleanupbot
PR: 13.33, ID:4875, Name:earle_martin
PR: 12.87, ID:6946, Name:useight
PR: 12.69, ID:3352, Name:crzrussian
```

Although we do not know how Wikipedia administrators are selected, after checking their usernames, those we checked were all engaged in Wikipedia administration during 2008. This shows that distributed computing is not just an airy idea in papers or academics, but can also work well in solving practical problems.

All necessary files and source code are put in the `PageRank` folder.

VI. Conclusion

Omitted.

References

[1] C. Li *et al.*, "Cloud computing course experiment handbook," 2021.

[2] K. Klausen *et al.*, "Arch linux cloud image build script," 2022. [Online]. Available: https://gitlab.archlinux.org/archlinux/arch-boxes

[3] C. Severance *et al.*, "Arch linux packaging script for hadoop," 2021. [Online]. Available: https://aur.archlinux.org/packages/hadoop

[4] huaweicloud.com, "Ims documentation," 2021. [Online]. Available: https://support.huaweicloud.com/ims/index.html

[5] Apache Software Foundation, "Hadoop documentation: Hadoop cluster setup," 2021. [Online]. Available: https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/ClusterSetup.html

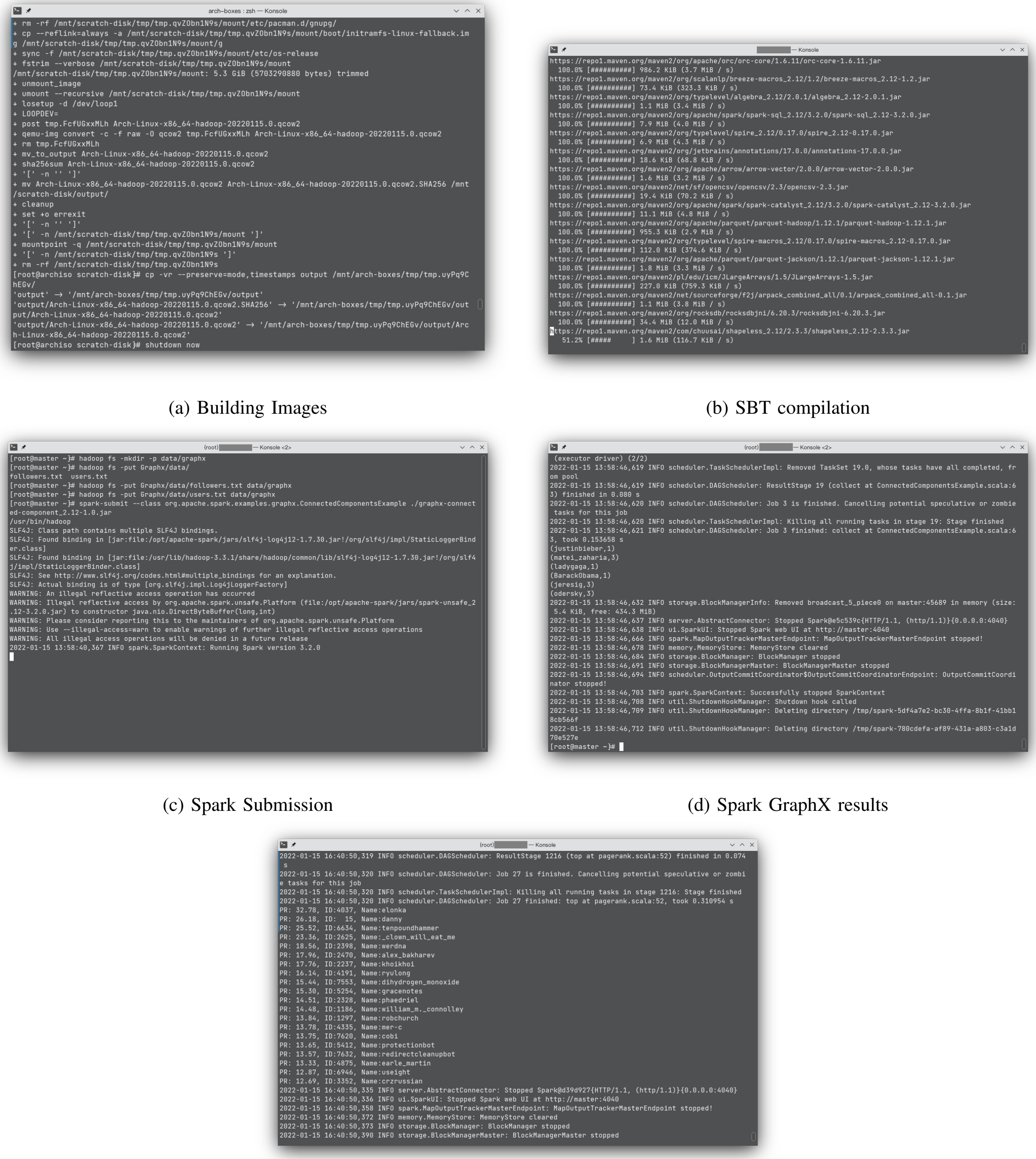
[6] Z. Lin, "Wordcount tutorial," 2017. [Online]. Available: http://dblab.xmu.edu.cn/blog/1311-2/

[7] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2010, pp. 1361–1370. [Online]. Available: http://snap.stanford.edu/data/wiki-Vote.html

[8] Apache Software Foundation, "Graphx programming guide," 2021. [Online]. Available: https://spark.apache.org/docs/latest/graphx-programming-guide.html

[9] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2010, pp. 1361–1370. [Online]. Available: http://snap.stanford.edu/data/wiki-Elec.html

Supplementary


arch–boxes : zsh — Konsole
[redacted] — Konsole
(root) [redacted] — Konsole <2>
(root) [redacted] — Konsole <2>
(root) [redacted] — Konsole

```
+ rm -rf /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount/etc/pacman.d/gnupg/
+ cp --reflink=always -a /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount/boot/initramfs-linux-fallback.im
g /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount/g
+ sync -f /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount/etc/os-release
+ fstrim --verbose /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount
/mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount: 5.3 GiB (5703290880 bytes) trimmed
+ unmount_image
+ umount --recursive /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount
+ losetup -d /dev/loop1
+ LOOPDEV=
+ post tmp.FcfUGxxMLh Arch-Linux-x86_64-hadoop-20220115.0.qcow2
+ qemu-img convert -c -f raw -O qcow2 tmp.FcfUGxxMLh Arch-Linux-x86_64-hadoop-20220115.0.qcow2
+ rm tmp.FcfUGxxMLh
+ mv_to_output Arch-Linux-x86_64-hadoop-20220115.0.qcow2
+ sha256sum Arch-Linux-x86_64-hadoop-20220115.0.qcow2
+ '[' -n '' ']'
+ mv Arch-Linux-x86_64-hadoop-20220115.0.qcow2 Arch-Linux-x86_64-hadoop-20220115.0.qcow2.SHA256 /mnt
/scratch-disk/output/
+ cleanup
+ set +o errexit
+ '[' -n '' ']'
+ '[' -n /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount ']'
+ mountpoint -q /mnt/scratch-disk/tmp/tmp.qvZObn1N9s/mount
+ '[' -n /mnt/scratch-disk/tmp/tmp.qvZObn1N9s ']'
+ rm -rf /mnt/scratch-disk/tmp/tmp.qvZObn1N9s
[root@archiso scratch-disk]# cp -vr --preserve=mode,timestamps output /mnt/arch-boxes/tmp/tmp.uyPq9C
hEGv/
'output' → '/mnt/arch-boxes/tmp/tmp.uyPq9ChEGv/output'
'output/Arch-Linux-x86_64-hadoop-20220115.0.qcow2.SHA256' → '/mnt/arch-boxes/tmp/tmp.uyPq9ChEGv/out
put/Arch-Linux-x86_64-hadoop-20220115.0.qcow2'
'output/Arch-Linux-x86_64-hadoop-20220115.0.qcow2' → '/mnt/arch-boxes/tmp/tmp.uyPq9ChEGv/output/Arc
h-Linux-x86_64-hadoop-20220115.0.qcow2'
[root@archiso scratch-disk]# shutdown now
```

(a) Building Images

```
https://repo1.maven.org/maven2/org/apache/orc/orc-core/1.6.11/orc-core-1.6.11.jar
  100.0% [##########] 986.2 KiB (3.7 MiB / s)
https://repo1.maven.org/maven2/org/scalanlp/breeze-macros_2.12/1.2/breeze-macros_2.12-1.2.jar
  100.0% [##########] 73.4 KiB (323.3 KiB / s)
https://repo1.maven.org/maven2/org/typelevel/algebra_2.12/2.0.1/algebra_2.12-2.0.1.jar
  100.0% [##########] 1.1 MiB (3.4 MiB / s)
https://repo1.maven.org/maven2/org/apache/spark/spark-sql_2.12/3.2.0/spark-sql_2.12-3.2.0.jar
  100.0% [##########] 7.9 MiB (4.0 MiB / s)
https://repo1.maven.org/maven2/org/typelevel/spire_2.12/0.17.0/spire_2.12-0.17.0.jar
  100.0% [##########] 6.9 MiB (4.3 MiB / s)
https://repo1.maven.org/maven2/org/jetbrains/annotations/17.0.0/annotations-17.0.0.jar
  100.0% [##########] 18.6 KiB (68.8 KiB / s)
https://repo1.maven.org/maven2/org/apache/arrow/arrow-vector/2.0.0/arrow-vector-2.0.0.jar
  100.0% [##########] 1.6 MiB (3.2 MiB / s)
https://repo1.maven.org/maven2/net/sf/opencsv/opencsv/2.3/opencsv-2.3.jar
  100.0% [##########] 19.4 KiB (70.2 KiB / s)
https://repo1.maven.org/maven2/org/apache/spark/spark-catalyst_2.12/3.2.0/spark-catalyst_2.12-3.2.0.jar
  100.0% [##########] 11.1 MiB (4.8 MiB / s)
https://repo1.maven.org/maven2/org/apache/parquet/parquet-hadoop/1.12.1/parquet-hadoop-1.12.1.jar
  100.0% [##########] 955.3 KiB (2.9 MiB / s)
https://repo1.maven.org/maven2/org/typelevel/spire-macros_2.12/0.17.0/spire-macros_2.12-0.17.0.jar
  100.0% [##########] 112.0 KiB (374.6 KiB / s)
https://repo1.maven.org/maven2/org/apache/parquet/parquet-jackson/1.12.1/parquet-jackson-1.12.1.jar
  100.0% [##########] 1.8 MiB (3.3 MiB / s)
https://repo1.maven.org/maven2/pl/edu/icm/JLargeArrays/1.5/JLargeArrays-1.5.jar
  100.0% [##########] 227.0 KiB (759.3 KiB / s)
https://repo1.maven.org/maven2/net/sourceforge/f2j/arpack_combined_all/0.1/arpack_combined_all-0.1.jar
  100.0% [##########] 1.1 MiB (3.8 MiB / s)
https://repo1.maven.org/maven2/org/rocksdb/rocksdbjni/6.20.3/rocksdbjni-6.20.3.jar
  100.0% [##########] 34.4 MiB (12.0 MiB / s)
https://repo1.maven.org/maven2/com/chuusai/shapeless_2.12/2.3.3/shapeless_2.12-2.3.3.jar
   51.2% [#####     ] 1.6 MiB (116.7 KiB / s)
```

(b) SBT compilation

```
[root@master ~]# hadoop fs -mkdir -p data/graphx
[root@master ~]# hadoop fs -put Graphx/data/
followers.txt  users.txt
[root@master ~]# hadoop fs -put Graphx/data/followers.txt data/graphx
[root@master ~]# hadoop fs -put Graphx/data/users.txt data/graphx
[root@master ~]# spark-submit --class org.apache.spark.examples.graphx.ConnectedComponentsExample ./graphx-connect
ed-component_2.12-1.0.jar
/usr/bin/hadoop
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/apache-spark/jars/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBind
er.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop-3.3.1/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4
j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/opt/apache-spark/jars/spark-unsafe_2
.12-3.2.0.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2022-01-15 13:58:40,367 INFO spark.SparkContext: Running Spark version 3.2.0
```

(c) Spark Submission

```
 (executor driver) (2/2)
2022-01-15 13:58:46,619 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 19.0, whose tasks have all completed, fr
om pool
2022-01-15 13:58:46,619 INFO scheduler.DAGScheduler: ResultStage 19 (collect at ConnectedComponentsExample.scala:6
3) finished in 0.080 s
2022-01-15 13:58:46,620 INFO scheduler.DAGScheduler: Job 3 is finished. Cancelling potential speculative or zombie
 tasks for this job
2022-01-15 13:58:46,620 INFO scheduler.TaskSchedulerImpl: Killing all running tasks in stage 19: Stage finished
2022-01-15 13:58:46,621 INFO scheduler.DAGScheduler: Job 3 finished: collect at ConnectedComponentsExample.scala:6
3, took 0.153658 s
(justinbieber,1)
(matei_zaharia,3)
(ladygaga,1)
(BarackObama,1)
(jeresig,3)
(odersky,3)
2022-01-15 13:58:46,632 INFO storage.BlockManagerInfo: Removed broadcast_5_piece0 on master:45689 in memory (size:
 5.4 KiB, free: 434.3 MiB)
2022-01-15 13:58:46,637 INFO server.AbstractConnector: Stopped Spark@e5c539c{HTTP/1.1, (http/1.1)}{0.0.0.0:4040}
2022-01-15 13:58:46,638 INFO ui.SparkUI: Stopped Spark web UI at http://master:4040
2022-01-15 13:58:46,666 INFO spark.MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
2022-01-15 13:58:46,678 INFO memory.MemoryStore: MemoryStore cleared
2022-01-15 13:58:46,684 INFO storage.BlockManager: BlockManager stopped
2022-01-15 13:58:46,691 INFO storage.BlockManagerMaster: BlockManagerMaster stopped
2022-01-15 13:58:46,694 INFO scheduler.OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordi
nator stopped!
2022-01-15 13:58:46,703 INFO spark.SparkContext: Successfully stopped SparkContext
2022-01-15 13:58:46,708 INFO util.ShutdownHookManager: Shutdown hook called
2022-01-15 13:58:46,709 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-5df4a7e2-bc30-4ffa-8b1f-41bb1
8cb566f
2022-01-15 13:58:46,712 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-780cdefa-af89-431a-a803-c3a1d
70e527e
[root@master ~]#
```

(d) Spark GraphX results

```
2022-01-15 16:40:50,319 INFO scheduler.DAGScheduler: ResultStage 1216 (top at pagerank.scala:52) finished in 0.074
 s
2022-01-15 16:40:50,320 INFO scheduler.DAGScheduler: Job 27 is finished. Cancelling potential speculative or zombi
e tasks for this job
2022-01-15 16:40:50,320 INFO scheduler.TaskSchedulerImpl: Killing all running tasks in stage 1216: Stage finished
2022-01-15 16:40:50,320 INFO scheduler.DAGScheduler: Job 27 finished: top at pagerank.scala:52, took 0.310954 s
PR: 32.78, ID:4037, Name:elonka
PR: 26.18, ID:  15, Name:danny
PR: 25.52, ID:6634, Name:tenpoundhammer
PR: 23.36, ID:2625, Name:_clown_will_eat_me
PR: 18.56, ID:2398, Name:werdna
PR: 17.96, ID:2470, Name:alex_bakharev
PR: 17.76, ID:2237, Name:khoikhoi
PR: 16.14, ID:4191, Name:ryulong
PR: 15.44, ID:7553, Name:dihydrogen_monoxide
PR: 15.30, ID:5254, Name:gracenotes
PR: 14.51, ID:2328, Name:phaedriel
PR: 14.48, ID:1186, Name:william_m._connolley
PR: 13.84, ID:1297, Name:robchurch
PR: 13.78, ID:4335, Name:mer-c
PR: 13.75, ID:7620, Name:cobi
PR: 13.65, ID:5412, Name:protectionbot
PR: 13.57, ID:7632, Name:redirectcleanupbot
PR: 13.33, ID:4875, Name:earle_martin
PR: 12.87, ID:6946, Name:useight
PR: 12.69, ID:3352, Name:crzrussian
2022-01-15 16:40:50,335 INFO server.AbstractConnector: Stopped Spark@d39d927{HTTP/1.1, (http/1.1)}{0.0.0.0:4040}
2022-01-15 16:40:50,336 INFO ui.SparkUI: Stopped Spark web UI at http://master:4040
2022-01-15 16:40:50,358 INFO spark.MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
2022-01-15 16:40:50,372 INFO memory.MemoryStore: MemoryStore cleared
2022-01-15 16:40:50,373 INFO storage.BlockManager: BlockManager stopped
2022-01-15 16:40:50,390 INFO storage.BlockManagerMaster: BlockManagerMaster stopped
```

(e) Page Rank results

Fig. 4: Supplementary Figures