雲計算和大數據入門

這節課包含以下話題：

- Spark
- Hadoop
- Kubernetes
- Docker
- Flink
- MongoDB

說起雲計算，似乎離不開很多的工具，Hadoop、Hive、Hbase、ZooKeeper、Docker、Kubernetes、Spark、Kafka、MongoDB、Flink、Druid、Presto、Kylin、Elastic Search。都有聽過嗎。這些工具有些我是從大數據工程師、分佈式後端工程師的職位描述上找到的。這些都是高薪職位。我們試著把他們都安裝上，試著把玩兩下。## 初探 Spark

官網說，`Spark` 用來處理大規模數據的分析引擎。`spark` 就是一套庫。它似乎不像 `Redis` 那樣分成服務端和客戶端。`spark` 就是只在客戶端使用的。從官網下載了最新的版本，`spark-3.1.1-bin-hadoop3.2.tar`。

```
$ tree . -L 1
.
  LICENSE
  NOTICE
  R
  README.md
  RELEASE
  bin
  conf
  data
  examples
  jars
  kubernetes
  licenses
  python
  sbin
  yarn
```

```
11 directories, 4 files
```

似乎就是各語言編寫的一些分析庫。

同時官網說可以在 Python 上直接裝依賴庫。`pip install pyspark`

```
$ pip install pyspark
Collecting pyspark
  Downloading pyspark-3.1.1.tar.gz (212.3 MB)
     |                 | 212.3 MB 14 kB/s
Collecting py4j==0.10.9
  Downloading py4j-0.10.9-py2.py3-none-any.whl (198 kB)
     |                 | 198 kB 145 kB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.1.1-py2.py3-none-any.whl size=212767604 sha256=0b8079e8
  Stored in directory: /Users/lzw/Library/Caches/pip/wheels/23/bf/e9/9f3500437422e2ab82246f25a51ee480a4
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9 pyspark-3.1.1
```

裝上了。

這會看官網，有些例子

```
./bin/run-example SparkPi 10
```

哦，原來可以運行剛剛下載的安裝包裡的程序。但出錯了。

```
$ ./bin/run-example SparkPi 10
21/03/11 00:06:15 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using
21/03/11 00:06:16 INFO ResourceUtils: No custom resources configured for spark.driver.
21/03/11 00:06:16 WARN Utils: Service 'sparkDriver' could not bind on a random free port. You may check
```

> Spark is a fast and general processing engine compatible with Hadoop data. It can run in Hadoop clusters through YARN or Spark’s standalone mode, and it can process data in HDFS, HBase, Cassandra, Hive, and any Hadoop InputFormat. It is designed to perform both batch processing (similar to MapReduce) and new workloads like streaming, interactive queries, and machine learning.

出現了好幾次 hadoop。谷歌了 spark depends hadoop 之後，找到這樣一段話。看來這依賴於 Hadoop 格式的數據。讓我們先研究 Hadoop。

Hadoop

簡單看了官網後。來安裝一下。

```
brew install hadoop
```

安裝的過程中，來了解一下。

> The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

就是說 Hadoop 是一套框架，來處理分佈式的數據集。這些數據集可能分部在很多計算機上。用很簡單的編程模型來處理。它是設計來從單一服務器擴展到千台機器的。與其依賴於硬件的高可用，這個庫則設計來在應用層就能檢查和處理錯誤。因此能將高可用的服務部署到集群中，雖然集群中的每台電腦都可能導致失敗。

```
$ brew install hadoop
Error:
  homebrew-core is a shallow clone.
  homebrew-cask is a shallow clone.
To `brew update`, first run:
  git -C /usr/local/Homebrew/Library/Taps/homebrew/homebrew-core fetch --unshallow
  git -C /usr/local/Homebrew/Library/Taps/homebrew/homebrew-cask fetch --unshallow
These commands may take a few minutes to run due to the large size of the repositories.
This restriction has been made on GitHub's request because updating shallow
clones is an extremely expensive operation due to the tree layout and traffic of
Homebrew/homebrew-core and Homebrew/homebrew-cask. We don't do this for you
automatically to avoid repeatedly performing an expensive unshallow operation in
CI systems (which should instead be fixed to not use shallow clones). Sorry for
```

```
the inconvenience!
==> Downloading https://homebrew.bintray.com/bottles/openjdk-15.0.1.big_sur.bottle.tar.gz
Already downloaded: /Users/lzw/Library/Caches/Homebrew/downloads/d1e3ece4af1d225bc2607eaa4ce85a873d2c6d
==> Downloading https://www.apache.org/dyn/closer.lua?path=hadoop/common/hadoop-3.3.0/hadoop-3.3.0.tar.
Already downloaded: /Users/lzw/Library/Caches/Homebrew/downloads/764c6a0ea7352bb8bb505989feee1b36dc628c
==> Installing dependencies for hadoop: openjdk
==> Installing hadoop dependency: openjdk
==> Pouring openjdk-15.0.1.big_sur.bottle.tar.gz
==> Caveats
For the system Java wrappers to find this JDK, symlink it with
  sudo ln -sfn /usr/local/opt/openjdk/libexec/openjdk.jdk /Library/Java/JavaVirtualMachines/openjdk.jdk

openjdk is keg-only, which means it was not symlinked into /usr/local,
because it shadows the macOS `java` wrapper.

If you need to have openjdk first in your PATH run:
  echo 'export PATH="/usr/local/opt/openjdk/bin:$PATH"' >> /Users/lzw/.bash_profile

For compilers to find openjdk you may need to set:
  export CPPFLAGS="-I/usr/local/opt/openjdk/include"

==> Summary
   /usr/local/Cellar/openjdk/15.0.1: 614 files, 324.9MB
==> Installing hadoop
   /usr/local/Cellar/hadoop/3.3.0: 21,819 files, 954.7MB, built in 2 minutes 15 seconds
==> Upgrading 1 dependent:
maven 3.3.3 -> 3.6.3_1
==> Upgrading maven 3.3.3 -> 3.6.3_1
==> Downloading https://www.apache.org/dyn/closer.lua?path=maven/maven-3/3.6.3/binaries/apache-maven-3.
==> Downloading from https://mirror.olnevhost.net/pub/apache/maven/maven-3/3.6.3/binaries/apache-maven-
######################################################################## 100.0%
Error: The `brew link` step did not complete successfully
The formula built, but is not symlinked into /usr/local
Could not symlink bin/mvn
Target /usr/local/bin/mvn
is a symlink belonging to maven. You can unlink it:
```

```
  brew unlink maven

To force the link and overwrite all conflicting files:
  brew link --overwrite maven

To list all files that would be deleted:
  brew link --overwrite --dry-run maven

Possible conflicting files are:
/usr/local/bin/mvn -> /usr/local/Cellar/maven/3.3.3/bin/mvn
/usr/local/bin/mvnDebug -> /usr/local/Cellar/maven/3.3.3/bin/mvnDebug
/usr/local/bin/mvnyjp -> /usr/local/Cellar/maven/3.3.3/bin/mvnyjp
==> Summary
   /usr/local/Cellar/maven/3.6.3_1: 87 files, 10.7MB, built in 7 seconds
Removing: /usr/local/Cellar/maven/3.3.3... (92 files, 9MB)
==> Checking for dependents of upgraded formulae...
==> No broken dependents found!
==> Caveats
==> openjdk
For the system Java wrappers to find this JDK, symlink it with
  sudo ln -sfn /usr/local/opt/openjdk/libexec/openjdk.jdk /Library/Java/JavaVirtualMachines/openjdk.jdk

openjdk is keg-only, which means it was not symlinked into /usr/local,
because it shadows the macOS `java` wrapper.

If you need to have openjdk first in your PATH run:
  echo 'export PATH="/usr/local/opt/openjdk/bin:$PATH"' >> /Users/lzw/.bash_profile

For compilers to find openjdk you may need to set:
  export CPPFLAGS="-I/usr/local/opt/openjdk/include"
```

注意到 `brew` 的輸出日誌中 `maven` 沒有很好地被鏈接。接下來，進行強制鏈接到 `3.6.3_1` 版本。

```
  brew link --overwrite maven
```

Hadoop 就安裝成功了。

Modules

The project includes these modules:

- **Hadoop Common**: The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS™)**: A distributed file system that provides high-throughput access to application data.
- **Hadoop YARN**: A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce**: A YARN-based system for parallel processing of large data sets.
- **Hadoop Ozone**: An object store for Hadoop.

說有這些模塊。這會敲入 `hadoop` 出現了：

```
$ hadoop
Usage: hadoop [OPTIONS] SUBCOMMAND [SUBCOMMAND OPTIONS]
 or    hadoop [OPTIONS] CLASSNAME [CLASSNAME OPTIONS]
  where CLASSNAME is a user-provided Java class

  OPTIONS is none or any of:

--config dir                     Hadoop config directory
--debug                          turn on shell script debug mode
--help                           usage information
buildpaths                       attempt to add class files from build tree
hostnames list[,of,host,names]   hosts to use in slave mode
hosts filename                   list of hosts to use in slave mode
loglevel level                   set the log4j level for this command
workers                          turn on worker mode

  SUBCOMMAND is one of:
    Admin Commands:

daemonlog     get/set the log level for each daemon

    Client Commands:

archive       create a Hadoop archive
checknative   check native Hadoop and compression libraries availability
```

```
classpath     prints the class path needed to get the Hadoop jar and the required libraries
conftest      validate configuration XML files
credential    interact with credential providers
distch        distributed metadata changer
distcp        copy file or directories recursively
dtutil        operations related to delegation tokens
envvars       display computed Hadoop environment variables
fs            run a generic filesystem user client
gridmix       submit a mix of synthetic job, modeling a profiled from production load
jar <jar>     run a jar file. NOTE: please use "yarn jar" to launch YARN applications, not this command
jnipath       prints the java.library.path
kdiag         Diagnose Kerberos Problems
kerbname      show auth_to_local principal conversion
key           manage keys via the KeyProvider
rumenfolder   scale a rumen input trace
rumentrace    convert logs into a rumen trace
s3guard       manage metadata on S3
trace         view and modify Hadoop tracing settings
version       print the version

    Daemon Commands:

kms           run KMS, the Key Management Server
registrydns   run the registry DNS server

SUBCOMMAND may print help when invoked w/o parameters or with -h.
```

官網給了些例子。

```
$ mkdir input
$ cp etc/hadoop/*.xml input
$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.2.jar grep input output 'dfs[a-z
$ cat output/*
```

注意到有 `share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.2.jar`。這意味著也許有些樣例文件我們沒有得到。猜測用 `Homebrew` 安裝會沒有這些文件。我們從官網下載了安裝文件包。

```
$ tree . -L 1
```

.

```
LICENSE-binary
LICENSE.txt
NOTICE-binary
NOTICE.txt
README.txt
bin
etc
include
lib
libexec
licenses-binary
sbin
share
```

出現了 `share` 目錄。然而 `Homebrew` 真的沒有附加的這些文件嗎。找到 `Homebrew` 安裝的目錄。

```
$ type hadoop
hadoop is /usr/local/bin/hadoop
$ ls -alrt /usr/local/bin/hadoop
lrwxr-xr-x  1 lzw  admin  33 Mar 11 00:48 /usr/local/bin/hadoop -> ../Cellar/hadoop/3.3.0/bin/hadoop
$ cd /usr/local/Cellar/hadoop/3.3.0
```

這是在`/usr/local/Cellar/hadoop/3.3.0/libexec/share/hadoop` 下打印的目錄樹

“ ‘shell $ tree . -L 2 . ├── client │ ├── hadoop-client-api-3.3.0.jar │ ├── hadoop-client-minicluster-3.3.0.jar │ └── hadoop-client-runtime-3.3.0.jar ├── common │ ├── hadoop-common-3.3.0-tests.jar │ ├── hadoop-common-3.3.0.jar │ ├── hadoop-kms-3.3.0.jar │ ├── hadoop-nfs-3.3.0.jar │ ├── hadoop-registry-3.3.0.jar │ ├── jdiff │ ├── lib │ ├── sources │ └── webapps ├── hdfs │ ├── hadoop-hdfs-3.3.0-tests.jar │ ├── hadoop-hdfs-3.3.0.jar │ ├── hadoop-hdfs-client-3.3.0-tests.jar │ ├── hadoop-hdfs-client-3.3.0.jar │ ├── hadoop-hdfs-httpfs-3.3.0.jar │ ├── hadoop-hdfs-native-client-3.3.0-tests.jar │ ├── hadoop-hdfs-native-client-3.3.0.jar │ ├── hadoop-hdfs-nfs-3.3.0.jar │ ├── hadoop-hdfs-rbf-3.3.0-tests.jar │ ├── hadoop-hdfs-rbf-3.3.0.jar │ ├── jdiff │ ├── lib │ ├── sources │ └── webapps ├── mapreduce │ ├── hadoop-mapreduce-client-app-3.3.0.jar │ ├── hadoop-mapreduce-client-common-3.3.0.jar │ ├── hadoop-mapreduce-client-core-3.3.0.jar │ ├── hadoop-mapreduce-client-hs-3.3.0.jar │ ├── hadoop-mapreduce-client-hs-plugins-3.3.0.jar │ ├── hadoop-mapreduce-client-jobclient-3.3.0-tests.jar │ ├──

hadoop-mapreduce-client-jobclient-3.3.0.jar │ ├── hadoop-mapreduce-client-nativetask-3.3.0.jar
│ ├── hadoop-mapreduce-client-shuffle-3.3.0.jar │ ├── hadoop-mapreduce-client-uploader-
3.3.0.jar │ ├── hadoop-mapreduce-examples-3.3.0.jar │ ├── jdiff │ ├── lib-examples
│ └── sources ├── tools │ ├── dynamometer │ ├── lib │ ├── resource