*** 請大家用以下的格式分享共筆:

==========

日期 ~自己的名字:

內容內容內容內容內容內容

~其他人的名字:

討論討論討論

==========

3/8 廖宜劭:

認識大數據的黃色小象幫手 — Hadoop

https://www.inside.com.tw/2015/03/12/big-data-4-hadoop

要懂 Hadoop，你必須先了解它最主要的兩項功能:

1. Hadoop 如何儲存資料(Store)
2. Hadoop 怎麼處理資料(Process)

這篇網址簡單介紹 Hadoop，並說明儲存資料的方法:分散式檔案系統 HDFS，以及處理資料的方法: MapReduce 平行運算架構。

==========

20180310~李書緯 :

HDFS introduction :

對於 HDFS 檔案系統架構中的各個元件進行簡單明瞭的介紹。

03/09 課堂中提到的 secondary namenode 並不提供 namenode 的冗餘(redundancy)、備援(backup)功能。Secondary namenode 主要的工作是把 namenode 當前的狀態快照(snapshot)以及更新的紀錄(log)定期整合起來，形成一個新的狀態快照，像是玩遊戲時會定期存檔，更新、保存當前的最新進度一樣。當 namenode 重啟時，即可用較快的速度恢復到最新或較新的狀態，一方面減輕 namenode 的工作負擔，另一方面同時提供狀態備份的功能。

[ref.]

http://limitedcode.blogspot.tw/2014/10/hdfs-hadoop-distributed-file-system-hdfs.html

HDFS command :

[ref.] https://www.tutorialspoint.com/hadoop/hadoop_command_reference.htm

==========

3/16 楊博鈞

有鑒於在上課的時候不是說很懂Hadoop，因此課後自己去大概小小的了解一下。

Hadoop: 有著 儲存、運算和資源管理功能的分散式Big Data處理平臺，提供三大項服務，也就是上課都有提到的。

1. HDFS:

儲存檔案到HDFS前，檔案會被拆開成數等分小區塊，稱之block，並且會將同一個block複製成數等分(replication, 預設值是3份)再將這些block分散儲存到各個DataNode，同時會產生一份清單，記載著這份檔案所屬的block與散落在哪幾台DataNode，這份清單會被記錄在NameNode上，而相同的block不會同時存在於同一個DataNode上。當某個Hadoop client需要讀取這個檔案時，會先跟NameNode發出請求，NameNode會根據這份清單回覆檔案的block位於哪幾台DataNode，Hadoop client再根據這份清單將各個block讀取出來，還原成一個完整個檔案。

NameNode：儲存檔案的block清單，稱之為metadata。
DataNode：負責儲存實體檔案的block。

2. Yarn:
   ResourceManager:主要是用來管理與裁決Hadoop叢集內資源的使用權
   NodeManager:是負責監控Hadoop叢集內每台機器的資源使用情況

當某個Job被submit到Yarn上面，Job會被拆成數個tasks並且產生一個ApplicationMaster (*AM*)，AM會負責與ResourceManager請求需要運算的資源，這時候ResourceManager會根據NodeManager回報的消息，告知AM哪幾台機器有空閒的資源可以使用

3. MapReduce：
   主要分為Map與Reduce兩個步驟。Map工作階段會把需要運算的資料拆分為多個獨立區塊(chunk)，平行運算完後第一階段的運算結果儲存於檔案系統上(通常會是在HDFS內)，進入Reduce階段會把Map運算的結果進行第二次的運算，運算出最後的結果。

參考資料：
https://ithelp.ithome.com.tw/articles/10190597

==========
3/16 ~方姿穎:

[上週課堂筆記與學習]
*Hadoop是什麼?
阿帕契管理開源的大數據分析軟體，任何人都可以在上面開發，以java編寫
特點:分散式、容錯、可擴充
*什麼情況下會用到Hadoop?
ETL(extract transform load),ex:收集逛過網站的資料，log的紀錄轉換成可分析的資料
Text mining(補充:與data mining不同之處?)
...(參考課程講義)
*core Hadoop
1.storage(HDFS)
2.resource management(YARN)看需要多少資源
3.processing(Spark MapReduce)
*獲取資料的工具
apache sqoop(把以前的資料丟進hadoop)
apache flume(固定觀察資料產生了沒，然後把它丟進hdfs)
*Hadoop漸漸被spark取代
hadoop mapreduce是primarily java-based，較為嚴謹，資料會落地做disk的存取。spark大部分在memory做存取（In-memory），相對沒那麼嚴謹，對開發人員來說較好操作。(了解更多)
*Hadoop cluster
master node(大腦)/worker node(做事)
最小儲存單位block(default 128MB)
資料備份(default *3)

name node 記錄資料儲存在哪個node, 時常使用可能會造成bottleneck
放置block的時候會考慮rack(機櫃)的位置, 同時兩個機櫃壞掉的機率很低
active name node/standby name node(狀況發生時接手工作)
active name node/secondary name node(幫忙處理metadata)

*HDFS command

(參考課程講義)
想檢查上傳的檔案, 可以用cat或是用瀏覽器開啟hostip:50070

*hadoop2.0

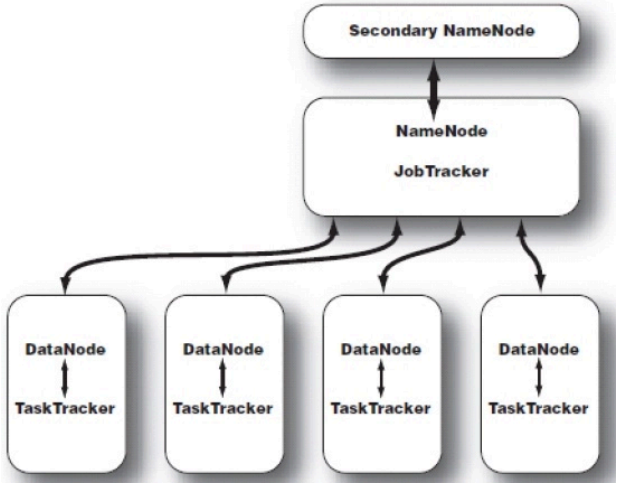
比喻:cluster飯店/rm櫃台/nm樓管/container房間(動態的)/am(領隊:負責check in)
(http://masterip:8088/cluster/apps 可以看cluster狀況)

==========

3/16 ~ 陳侑炘 :

Hardoop 基本架構

Hardoop 1.x



NameNode : HDFS的核心程序, 對整個分布式文件系統進行總控制, 會紀錄所有的元數據分布存儲的狀態信息(比如文件是如何分割成數據塊的, 以及這些數據塊被存儲到哪些節點上, 還有對內存和I/O進行集中管理)
但這是個單點, 發生故障會使集群崩潰!!! 故需要Secondary NameNode

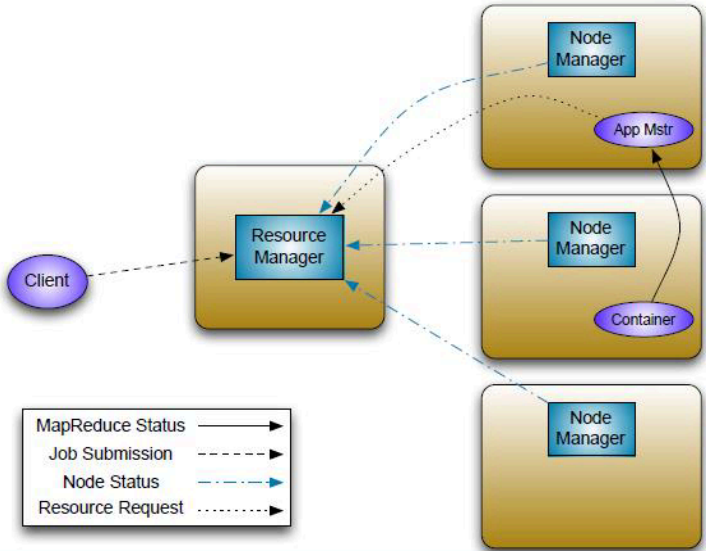
Secondary NameNode : 可想像是 → 輔助名稱節點 / 檢查節點, 它是監控HDFS狀態的輔助後台程序, 可以保存NameNode的副本, 故每個集群都有一個, 它與NameNode進行通訊, 定期保存HDFS元數據快照。
NameNode 發現有哪個 DataNode 上的資料遺失或遭到損壞, 就會尋找其他 DataNode 上的副本(Replica)進行複製, 保持整個系統都有三份的狀態。

DataNode : 數據節點, 負責把HDFS數據塊讀、寫到本地文件系統

MapReduce
JobTracker : 用於處理作業(用戶提交的代碼)的後台程序, 決定有哪些文件參與作業的處理, 然後把作業切割成為一個個的小task, 並把它們分配到所需要的數據所在的子節點。
→ 監控 task + 重啟失敗的 task

→ 位於Master節點，每個集群只有一個 !!!
TaskTracker： 與 DataNode 結合（代碼與數據一起的原則），管理各自節點上的task（由 jobtracker分配），用於並行執行 map 或 reduce任務，它與 jobtracker 交互通信，可以告知 jobtracker 子任務完成情況。
→ map : 在各個節點上處理資料片段，把工作分散、分佈出去的這個階段叫做 Mapping
→ reduce : 把各節點運算出的結果直接傳送回來歸納整合，這個階段就叫做 Reducing
→ 位於 Slave， 每個節點只有一個 tasktracker，但一個 tasktracker 可以啟動多個JVM

Hardoop 2.X
原有的JobTracker兩個主要功能 : 資源管理 + 任務調度/監控，而新版本將資源管理分離出來，Yarn 接替 TaskScheduler(資源管理) 的工作。



Yarn → RM (ResourseManager) & AM (ApplicationMaster)
RM : 系統將資源分配給各個應用的最終決策者
AM : 是一個具體的框架庫，任務是與RM協商獲取應用所需資源 + 與NM合作以完成執行和監控task 。
NM (Node Manager)： 每個節點一個，用於管理節點上的 task 和資源
Container : 容器。內有所需機器資源(CPU、磁碟、網路..等)，每個任務被分配一個container，並且該任務只能在該 container 中執行。

step 1 : Client 將需求提交到 RM
step 2 : RM 為 AM 申請資源，並和某個 NM 通信以啟動 AM
step 3 : AM 與 RM 開始通信，並為內部要執行的任務(task)申請資源，一旦得到資源後，將和 NM 通信以啟動對應的任務 (task)

參考資料 :
https://www.ibm.com/developerworks/cn/opensource/os-cn-hadoop-yarn/
http://www.bigdatafinance.tw/index.php/tech/methodology/409-hadoop
http://dongxicheng.org/mapreduce-nextgen/basic-concepts/

==========

3/22~許雅雯
[關於 hadoop2.7.5 及 spark 在本機端的運行的環境建置]
環境是 Ubuntu16.04
為了在自己電腦上也能運行 SPARK 將助教給的環境建制路徑都導到自己的檔案
跟助教給的檔案比較不一樣的是我的 YARN_CONF_DIR 設定，在 hadoop2.7.5 下並沒有找到 hadoop-conf
所以我馬上又去找 etc 底下的 hadoop 裏面有 env 和一些 conf 果然裏面有一些預設的路徑宣告，所以就把他導到 /etc/hadoop/

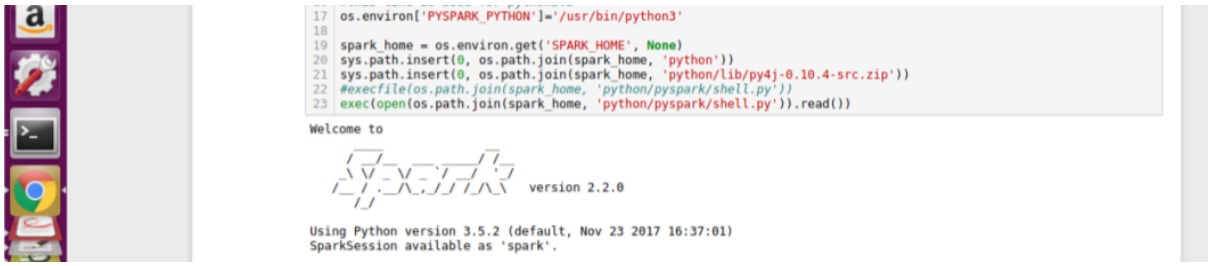
```
os.environ['JAVA_HOME']='/usr/lib/jvm/java-8-openjdk-amd64/'
os.environ['YARN_CONF_DIR']='/usr/local/hadoop/etc/hadoop/'

#this line is used for spark2.2
os.environ['SPARK_HOME']='/usr/local/spark'

#this line is used for python3.5
os.environ['PYSPARK_PYTHON']='/usr/bin/python3'
spark_home = os.environ.get('SPARK_HOME', None)
sys.path.insert(0, os.path.join(spark_home, 'python'))
sys.path.insert(0, os.path.join(spark_home, 'python/lib/py4j-0.10.4-src.zip'))
exec(open(os.path.join(spark_home, 'python/pyspark/shell.py')).read())
```

再來是 PYSPARK_SUBMIT_ARGS 的設定，如果像我這篇沒有設定，他的預設是 pyspark-shell 的執行檔
像是 jupyter_spark.py 裡使用的>>
--verbose 的作用：
If you are ever unclear where configuration options are coming from, you can print out fine-grained debugging information by running spark-submit with the --verbose option.

--master 的用法：
通常是放 The master URL for the cluster (e.g. spark://23.195.26.187:7077)
接 yarn 的話代表 Run on a YARN cluster
除此之外,也可設定 local[核心數]

參數的講解都可以在spark 網站上查詢
https://spark.apache.org/docs/latest/submitting-applications.html

設定完成後就可以再自己電腦上 jupyter_notebook 運行 spark，會有以下畫面：



==========