spark + R

### 발표 내용

spark + R 기본 사용법 특징과 장단점 소개

기존 스터디 내용 : <https://github.com/biospin/R_Bio>

### R에서 spark 을 연동 방법

* 1. SparkR (R on Spark) : <http://spark.apache.org/docs/latest/sparkr.html>
  2. sparklyr — R interface for Apache Spark : <http://spark.rstudio.com/>

### SparkR (R on Spark)의 설치와 사용법

* Windows 가능하지만 유닉스 계열(맥, 리눅스)이 더욱 쉬움.
* Bash on Ubuntu on Windows에서는 R-Studio Server가 동작 X
* CentOS 6.7에서 실행
* <http://spark.apache.org/downloads.html> 에서 spark-2.0.2-bin-hadoop2.7.tgz 다운로드후에 압축 풀기
* vi /etc/hosts 안에 hostname 이 꼭 등록 필요

wget http://d3kbcqa49mib13.cloudfront.net/spark-2.0.2-bin-hadoop2.7.tgz   
tar xvf spark-2.0.2-bin-hadoop2.7.tgz   
ln -s spark-2.0.2-bin-hadoop2.7 spark  
  
sudo vi /etc/hosts  
127.0.0.1 {myhostName}

* Spark에서 제공하는 R과 lib을 사용함.

if (nchar(Sys.getenv("SPARK\_HOME")) < 1) {  
 Sys.setenv(SPARK\_HOME = "/home/goodmit/spark")  
}  
library(SparkR, lib.loc = c(file.path(Sys.getenv("SPARK\_HOME"), "R", "lib")))

##   
## Attaching package: 'SparkR'

## The following objects are masked from 'package:stats':  
##   
## cov, filter, lag, na.omit, predict, sd, var, window

## The following objects are masked from 'package:base':  
##   
## as.data.frame, colnames, colnames<-, drop, endsWith,  
## intersect, rank, rbind, sample, startsWith, subset, summary,  
## transform, union

sparkR.session(master = "local[\*]",   
 sparkConfig = list(spark.driver.memory = "2g"),  
 sparkPackages = "com.databricks:spark-avro\_2.11:3.0.0" )

## Spark package found in SPARK\_HOME: /home/goodmit/spark

## Launching java with spark-submit command /home/goodmit/spark/bin/spark-submit --packages com.databricks:spark-avro\_2.11:3.0.0 --driver-memory "2g" sparkr-shell /tmp/RtmpzGLbcx/backend\_portdbb3b5884fd

## Java ref type org.apache.spark.sql.SparkSession id 1

* master = "local[\*]" 을 수정해서 spark cluster로 접속 가능
  + 예) master = "spark://xxx.xxx.xxx:2345"
  + 예) master = "yarn"
  + 예) master = "mesos://xxx.xxx.xxx:5050"

#### SparkDataFrames 생성

* From local data frames

df <- as.DataFrame(faithful)  
head(df)

## eruptions waiting  
## 1 3.600 79  
## 2 1.800 54  
## 3 3.333 74  
## 4 2.283 62  
## 5 4.533 85  
## 6 2.883 55

* From Data Sources

people <- read.df("/home/goodmit/spark/examples/src/main/resources/people.json", "json")  
head(people)

## age name  
## 1 NA Michael  
## 2 30 Andy  
## 3 19 Justin

printSchema(people)

## root  
## |-- age: long (nullable = true)  
## |-- name: string (nullable = true)

#### SparkDataFrame Operations

df

## SparkDataFrame[eruptions:double, waiting:double]

# Select only the "eruptions" column  
head(select(df, df$eruptions))

## eruptions  
## 1 3.600  
## 2 1.800  
## 3 3.333  
## 4 2.283  
## 5 4.533  
## 6 2.883

# Filter the SparkDataFrame to only retain rows with wait times shorter than 50 mins  
head(filter(df, df$waiting < 50))

## eruptions waiting  
## 1 1.750 47  
## 2 1.750 47  
## 3 1.867 48  
## 4 1.750 48  
## 5 2.167 48  
## 6 2.100 49

# We use the `n` operator to count the number of times each waiting time appears  
head(summarize(groupBy(df, df$waiting), count = n(df$waiting)))

## waiting count  
## 1 70 4  
## 2 67 1  
## 3 69 2  
## 4 88 6  
## 5 49 5  
## 6 64 4

# We can also sort the output from the aggregation to get the most common waiting times  
waiting\_counts <- summarize(groupBy(df, df$waiting), count = n(df$waiting))  
head(arrange(waiting\_counts, desc(waiting\_counts$count)))

## waiting count  
## 1 78 15  
## 2 83 14  
## 3 81 13  
## 4 77 12  
## 5 82 12  
## 6 79 10

# Convert waiting time from hours to seconds.  
df$waiting\_secs <- df$waiting \* 60  
head(df)

## eruptions waiting waiting\_secs  
## 1 3.600 79 4740  
## 2 1.800 54 3240  
## 3 3.333 74 4440  
## 4 2.283 62 3720  
## 5 4.533 85 5100  
## 6 2.883 55 3300

#### spark + R 의 단점

* R로 MapReduce방식의 코드 구현 예시
  + <https://github.com/biospin/R_Bio/blob/master/part02/week4_160920/SparkR_chap03.Top10List.ipynb>
* MapReduce방식의 코드와 dataframe을 사용할때 차이점
  + <https://github.com/biospin/R_Bio/blob/master/part03/week1_161004/sparkR/sparkR_chap04.LeftOuterJoin.ipynb>