CS 649 Big Data: Tools and Methods
Spring Semester, 2021
Doc 17 Spark Reminder
Mar 25, 2021

Understand This

```
counter = 0
def count(item):
    global counter
    print("item: ", item.id, "counter: ", counter)
    counter = counter + 1

from pyspark.sql import SparkSession
spark = SparkSession.builder \
        .appName("Counter") \
        .getOrCreate()
spark.sparkContext

print("start")
df = spark.range(16)
smaller = df.coalesce(4)
smaller.foreach(count)
print("end")
print(counter)
```

Output in Notebook

```
start
end
0
```

Output at Command line

```
item:  12 counter:  0
item:  13 counter:  1
item:  14 counter:  2
item:  15 counter:  3
item:  4 counter:  0
item:  5 counter:  1
item:  6 counter:  2
item:  7 counter:  3
item:  8 counter:  0
item:  9 counter:  1
item:  10 counter:  2
item:  11 counter:  3
item:  0 counter:  0
item:  1 counter:  1
item:  2 counter:  2
item:  3 counter:  3
```

Towards AWS

Need a program

Issues

- Packaging files
- Running in local cluster of one machine
- Logging
- File references

Sample Program

printExample.py

```
from __future__ import print_function
def print5000():
    from pyspark.sql import SparkSession
    spark = SparkSession.builder \
        .master("local") \
        .appName("Print") \
        .getOrCreate()
    print(spark.range(5000).selectExpr("sum(id)").collect())

if __name__ == "__main__":
        print5000()
```

Running in Temp Spark Runtime

I put the SPARK_HOME/bin & SPARK_HOME/sbin on my path

Set SPARK_HOME

setenv SPARK_HOME /Java/spark-3.1.1-bin-hadoop3.2

->spark-submit ./printExample.py

21/03/25 10:49:50 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform
using builtin-java classes where applicable

Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties

21/03/25 10:49:52 INFO SparkContext: Running Spark version 3.1.1

21/03/25 10:49:52 INFO ResourceUtils:

==========

21/03/25 10:49:52 INFO ResourceUtils: No custom resources configured for spark.driver.

21/03/25 10:49:52 INFO ResourceUtils:

==========

21/03/25 10:49:52 INFO SparkContext: Submitted application: Print

2

Output

71 lines

```
21/03/25 10:49:58 INFO TaskSetManager: Finished task 0.0 in stage 0.0 (TID 0) in 300 m
192.168.1.16 (executor driver) (1/1)
21/03/25 10:49:58 INFO TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks have a
from pool
21/03/25 10:49:58 INFO DAGScheduler: ResultStage 0 (collect at /Users/whitney/Course
Spring21/examples/spark/./printExample.py:8) finished in 0.470 s
21/03/25 10:49:58 INFO DAGScheduler: Job 0 is finished. Cancelling potential speculati
tasks for this job
21/03/25 10:49:58 INFO TaskSchedulerImpl: Killing all running tasks in stage 0: Stage fi
21/03/25 10:49:58 INFO DAGScheduler: Job 0 finished: collect at /Users/whitney/Course
Spring21/examples/spark/./printExample.py:8, took 0.505491 s
Line 59 → [Row(sum(id)=12497500)]
21/03/25 10:49:58 INFO SparkContext: Invoking stop() from shutdown hook
21/03/25 10:49:58 INFO SparkUI: Stopped Spark web UI at http://192.168.1.16:4040
21/03/25 10:49:58 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterE
stopped!
21/03/25 10:49:58 INFO MemoryStore: MemoryStore cleared
21/03/25 10:49:58 INFO BlockManager: BlockManager stopped
21/03/25 10:49:58 INFO BlockManagerMaster: BlockManagerMaster stopped
```

File input/output

Hardcoding I/O file names in source not desirable

parseExample.py

```
    def files_from_args():
        import argparse
        parser = argparse.ArgumentParser()
        parser.add_argument('-i', '--input', default='input')
        parser.add_argument('-o', '--output',default='output')
        args = parser.parse_args()
        return (args.input, args.output)

if __name__ == "__main__":
        inputfile, outputfile = files_from_args()
        print("input = " + inputfile)
        print("output = " + outputfile)
```

Example Usage

->python parseExample.py -i cat
input = cat
output = output

->python parseExample.py -i cat -output dog
usage: parseExample.py [-h] [-i INPUT] [-o OUTPUT]
printExample.py: error: unrecognized arguments: dog

->python parseExample.py -i cat --output dog
input = cat
output = dog

->python parseExample.py --output dog -i cat
input = cat
output = dog

Sample Program

```
def write5000(file):
    from pyspark.sql import SparkSession
    spark = SparkSession.builder \
        .appName("Write") \
        .getOrCreate()

    spark.range(5000).selectExpr('id *2').write.format('csv').save(file)
    spark.stop()
```

```
def files_from_args():
        import argparse
        parser = argparse.ArgumentParser()
        parser.add_argument('-i', '--input', default='input')
        parser.add_argument('-o', '--output',default='output')
        args = parser.parse_args()
        return (args.input, args.output)

if __name__ == "__main__":
        _, outputfile = files_from_args()
        write5000(outputfile)
```

->spark-submit ./writeExample.py

21/03/25 10:55:28 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties

21/03/25 10:55:29 INFO SparkContext: Running Spark version 3.1.1

21/03/25 10:55:29 INFO ResourceUtils:
==

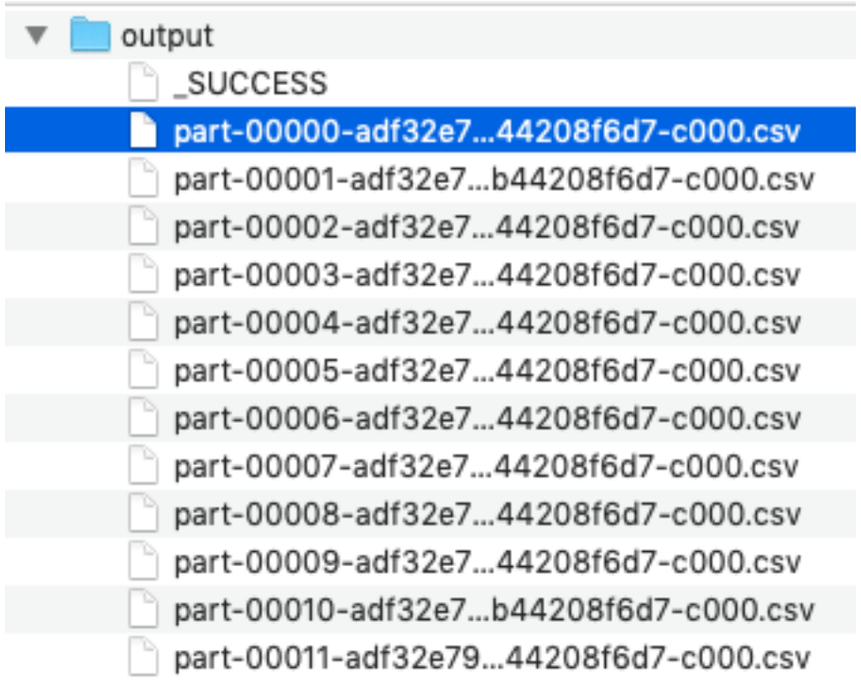
21/03/25 10:55:29 INFO ResourceUtils: No custom resources configured for spark.driver.

21/03/25 10:55:29 INFO ResourceUtils:
==

21/03/25 10:55:29 INFO SparkContext: Submitted application: Write

178 log lines sent to console

Output



part-00000	part-00001
0	832
2	834
4	836
6	838
8	840
10	842
...	...
812	1646
814	1648
816	1650
818	1652
820	1654
822	1656
824	1658
826	1660
828	1662
830	1664

Starting a Spark Cluster of One

Command SPARK_HOME/sbin/start-master.sh

>start-master.sh

starting org.apache.spark.deploy.master.Master, logging to /Java/spark-3.1.1-bin-hadoop3.2/logs/spark-whitney-org.apache.spark.deploy.master.Master-1-COS-CS-E052883.out

Master Web Page

localhost:8080

127.0.0.1:8080

0.0.0.0:8080

Spark 3.1.1 **Spark Master at spark://COS-CS-E052883:7077**

URL: spark://COS-CS-E052883:7077
Alive Workers: 0
Cores in use: 0 Total, 0 Used
Memory in use: 0.0 B Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (0)

Worker Id	Address	State	Cores	Memory	Resources

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Starting slave on local machine

Command SPARK_HOME/sbin/start-worker.sh

->start-worker.sh spark://COS-CS-E052883:7077

Master Web Page

Spark Master at spark://COS-CS-E052883:7077

URL: spark://COS-CS-E052883:7077
Alive Workers: 1
Cores in use: 12 Total, 0 Used
Memory in use: 15.0 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory	Resources
worker-20210325112451-192.168.1.16-61176	192.168.1.16:61176	ALIVE	12 (0 Used)	15.0 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Submitting Job to Spark on Cluster

run SPARK_HOME/bin/spark-submit

->spark-submit --master spark://rew-2.local:7077 ./printExample.py -o sampleOut

Master Web Page

Spark Master at spark://rew-2.local:7077

2.4.0

URL: spark://rew-2.local:7077
Alive Workers: 1
Cores in use: 8 Total, 0 Used
Memory in use: 15.0 GB Total, 0.0 B Used
Applications: 0 Running, 1 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory
worker-20190324215103-192.168.0.112-64770	192.168.0.112:64770	ALIVE	8 (0 Used)	15.0 GB (0.0 B Used)

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

Completed Applications (1)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
app-20190324215330-0000	Write	8	1024.0 MB	2019/03/24 21:53:30	whitney	FINISHED	6 s

Application Page

Application: Write

ID: app-20190324215330-0000
Name: Write
User: whitney
Cores: Unlimited (8 granted)
Executor Limit: Unlimited (1 granted)
Executor Memory: 1024.0 MB
Submit Date: 2019/03/24 21:53:30
State: FINISHED

Executor Summary (1)

ExecutorID	Worker	Cores	Memory	State	Logs

Removed Executors (1)

ExecutorID	Worker	Cores	Memory	State	Logs
0	worker-20190324215103-192.168.0.112-64770	8	1024	KILLED	stdout stderr

Starting/Stopping Master/Slave

Commands in SPARK_HOME/sbin

->start-master.sh

->start-slave.sh spark://air-6.local:7077

->stop-master.sh

->stop-slave.sh

->start-all.sh

->stop-all.sh

CAP Theorem

CAP theorem says in a distributed system you can not have all three

- Consistency
- Availability
- Tolerance to network Partitions

Consistency

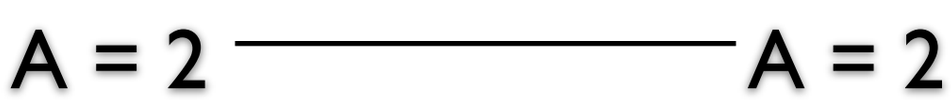
Machine 1 Machine 2

A = 2 —————————— A = 2

Not Consistent A = 2 A = 3

Partition

Machine 1 Machine 2



Partitioned

Machine 1 cannot

talk to machine 2

But how does machine 1 tell the difference between no connection and a very slow connection or busy machine 2?

Latency

Latency

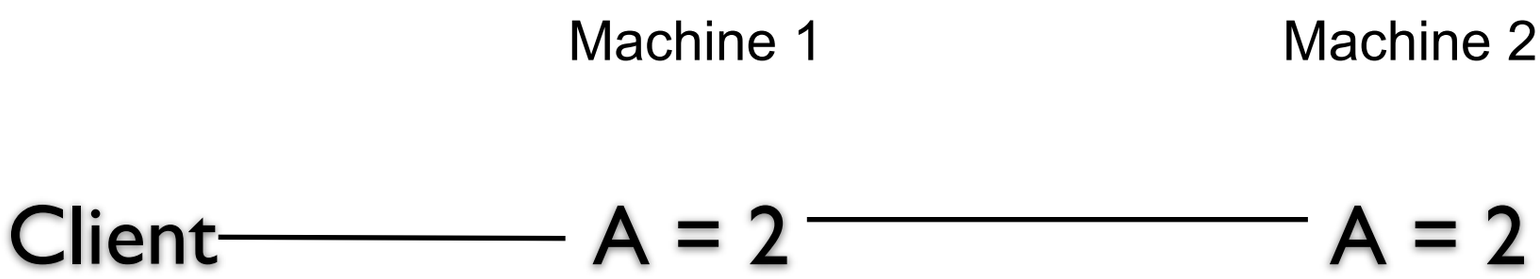
Time between making a request and getting a response

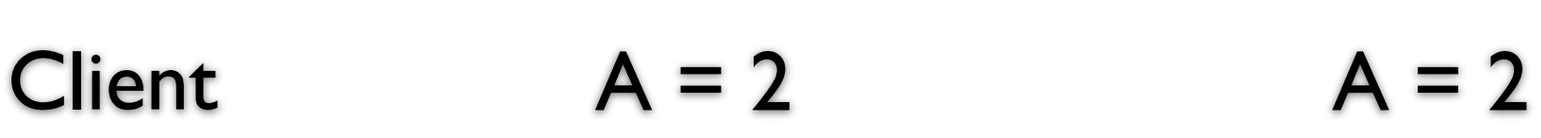
Distributed systems always have latency

In practice detect a partition by latency

When no response in a given time frame assume we are partitioned

Available





Client can not access value of A

What does not available mean?

- No connection
- Slow connection
- What is the difference?

Some say high available - meaning low latency

In practice available and latency are related

Consistency over Latency

Machine 1 Machine 2

Set A to 3 A = 2 A = 2

Set A to 3 A = 2 Lock A A = 2

Write requests queued until unlocked

Set A to 3 A = 2 Set A to 3 A = 2

Increased latency
System still available

Set A to 3 A = 3 Unlock A A = 3

A = 3 A = 3

Latency over Consistency

Machine 1

Machine 2

Set A to 3

A = 2

A = 2

A = 3

A = 2

Write requests accepted

Set A to 3

Low latency

System inconsistent

A = 3

A = 2

A = 3

A = 3

Latency over Consistency - Write Conflicts

Machine 1

Machine 2

Set A to 3

A = 2

A = 2

Subtract 1 from A

A = 3

A = 2

Set A to 3

A = 3

A = 1

Subtract 1 from A

A = ?

A = ?

Need policy to make system consistent

Partition

Machine 1	Machine 2
A = 2	A = 2
A = 2	A = 2
Set A to 3	Subtract 1 from A
A = 3	A = 1
A = ?	A = ?
	Need policy to make system consistent

CAP Theorem

Not a theorem

Too simplistic
- What is availability
- What is a partition of the network

Misleading

Intent of CAP was to focus designers attention on the tradeoffs in distributed systems

- How to handle partitions in the network
- Consistency
- Latency
- Availability

CAP & S3

S3 favors latency over consistency