



# Parallel streaming trace processing in Spark

Daniel Capelo Borges  
December 9, 2019

Pr. Michel Dagenais

Polytechnique Montréal  
Laboratoire **DORSAL**

# Agenda

---

- × What Apache Spark is ...
- × Cloud Platform
- × Research Challenges
- × Early results
- × Future work
- × References

# Agenda

---

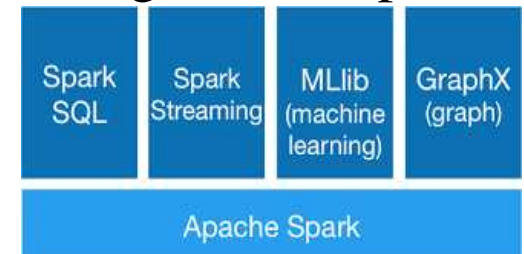
- × What Apache Spark is ...
- × Cloud Platform
- × Research Challenges
- × Early results
- × Future work
- × References

# What Apache Spark is... 1/4

---

## Introduction

- × Apache Spark is an **open source** cluster computing framework and it uses **memory operations** divided into several processing phases. Like **Apache Hadoop**, Apache Spark makes use of the MapReduce[1] programming model published by Google to perform distributed computing across several nodes.
- × Apache Spark works with batch and stream processing.
- × Apache Spark includes built-in libraries like Spark SQL (for working with structured data), Spark Streaming (for building scalable fault-tolerant streaming applications), MLlib (Machine Learning) and GraphX (for graphs and graph-parallel computation).



Spark libraries – from <https://spark.apache.org/>

# What Apache Spark is... 2/4

## × Some use cases...



Analyzing Log Data With Apache Spark

- <https://databricks.com/session/analyzing-log-data-with-apache-spark>



Netflix's Recommendation ML Pipeline Using Apache Spark

- <https://databricks.com/session/netflixs-recommendation-ml-pipeline-using-apache-spark>



Analyzing 2TB of Raw Trace Data from a Manufacturing Process: A First Use Case of Apache Spark for Semiconductor Wafers from Real Industry

- <https://databricks.com/session/analyzing-2tb-of-raw-trace-data-from-a-manufacturing-process-a-first-use-case-of-apache-spark-for-semiconductor-wafers-from-real-industry>

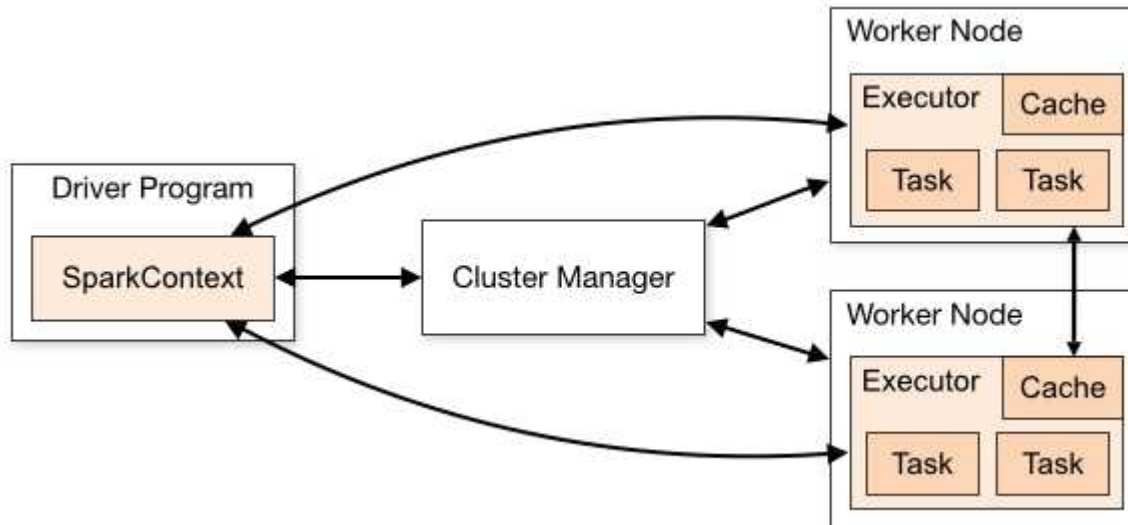


Automating Predictive Modeling at Zynga with PySpark and Pandas UDFs

- <https://databricks.com/session/automating-predictive-modeling-at-zynga-with-pyspark-and-pandas-udfs>

# What Apache Spark is... 3/4

## x Architecture

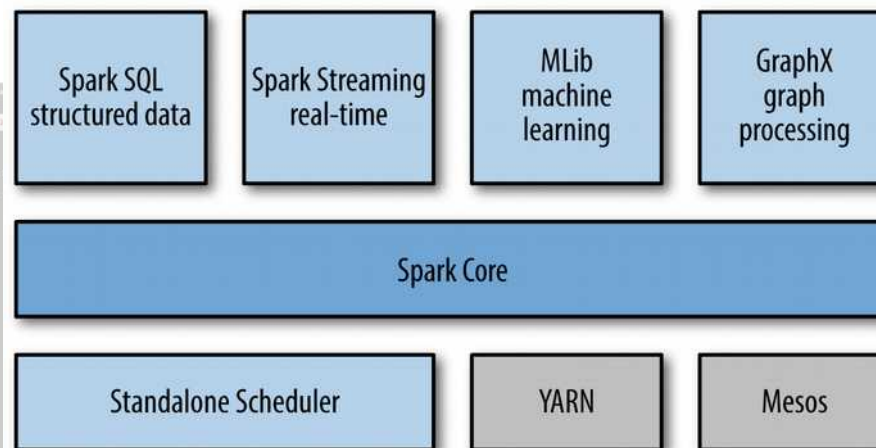


Spark architecture – from <https://spark.apache.org/docs/latest/cluster-overview.html>

# What Apache Spark is... 4/4

## Deployment

- × Apache Spark can run locally,
- × as a Standalone Cluster or on Hadoop YARN Cluster !



The Apache Spark stack from [2]

The screenshot shows a Spark cluster monitoring dashboard with a table of worker nodes. The table has columns for Name, Hostname, Address, Role, Status, Driver, and Memory. The data is as follows:

Name	Hostname	Address	Role	Status	Driver	Memory
spark-00000000000000000000	192.168.43.100	192.168.43.100:7077	Worker	Running	None	1024 MB
spark-00000000000000000001	192.168.43.101	192.168.43.101:7077	Worker	Running	None	1024 MB

# Agenda

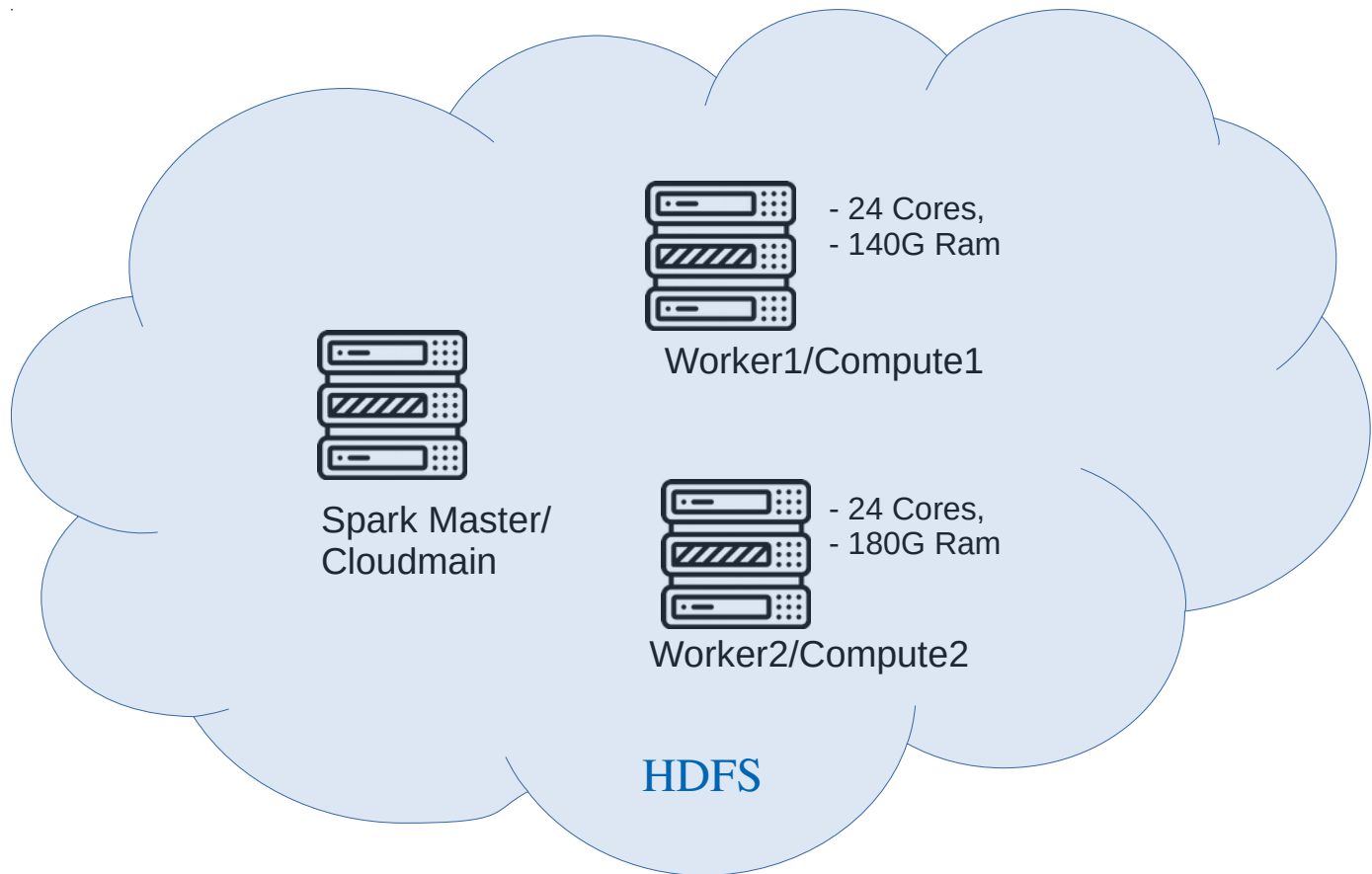
---

- × What Apache Spark is ...
- × **Cloud Platform**
- × Research Challenges
- × Early results
- × Future work
- × References



# Cloud Platform 1/1

---





# Agenda

---

- × What Apache Spark is ...
- × Cloud Platform
- × **Research Challenges**
- × Early results
- × Future work
- × References

# Research Challenges

---

- × How can we **speed up** trace analysis with Spark? 
  - × How can we import data (specifically **LTTng traces**) into Spark?
  - × How to filter and structure the data before further processing?
- × How to efficiently trace execution in a Spark environment? 
  - × What are the most useful metrics, analysis and views to develop to study Spark systems?
  - × How can we use **LTTng** to trace Spark?
- × How can we analyse traces in real-time/streaming with Spark?
  - × How can we use the Spark environment to analyse large/"infinite" traces being streamed out of a large system?

# Agenda

---

- x What Apache Spark is ...
- x Cloud Platform
- x Research Challenges
- x **Early results**
- x Future work
- x References



# Early results... 2/2

- ✓ Reproducing some papers (trace, logs and sentiment analysis; anomaly detection, etc).

```
Jupyter Proj1-SCADSReproduction Last Checkpoint: 22/10/2019 (autosaved)
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3
In [6]:
# Show DataFrame DFTraces and count it (number of lines).
# Each Line in DataFrame, it corresponds to a file.

print ("Training trace files imported into DataFrame - DFTracesTraining\n")
DFTracesTraining.show(5,truncate=50)
vocabSize = DFTracesTraining.count()
print ("Number of Training trace files: ", vocabSize) # number of files in the folder = number of lines in DataFrame

print ("\n\nValidation trace files imported into DataFrame - DFTracesValidation\n")
DFTracesValidation.show(5,truncate=50)
vocabSize = DFTracesValidation.count()
print ("Number of Validation trace files: ", vocabSize)

Training trace files imported into DataFrame - DFTracesTraining
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     | value|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|[142, 3, 78, 3, 4, 5, 221, 142, 4, 6, 142, 3, 1...|
|[311, 331, 168, 168, 42, 221, 221, 221, 221, 22...|
|[6, 11, 192, 6, 192, 125, 125, 6, 45, 45, 192, ...|
|[6, 54, 140, 221, 11, 5, 6, 125, 20, 5, 3, 6, 1...|
|[5, 63, 199, 174, 5, 102, 102, 168, 265, 265, 1...|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

Number of Training trace files: 833

Validation trace files imported into DataFrame - DFTracesValidation

In [7]:
# Algorithm 2 : Generate words of system calls with the single-length n-gram (1-gram) method from DataFrame DFTrace
#
```

```
Jupyter Proj1-SCADSReproduction Last Checkpoint: 22/10/2019 (autosaved)
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3
# Algorithm 2 : Generate words of system calls with the multiple-length n-gram (n-gram) method from DataFrame DFTrac
#
#
# https://spark.apache.org/docs/2.2.0/ml-features.html#n-gram
# https://spark.apache.org/docs/2.2.0/api/python/pyspark.ml.html#pyspark.ml.feature.NGram
#
# Using n=6
#
from pyspark.ml.feature import NGram

ngram = NGram(n=6, inputCol="value", outputCol="ngrams")

DFngramSix = ngram.transform(DFTracesTraining)
#DFngramSix.show(5)
DFngramSix.select("ngrams").show(5, truncate=50)
DFngramSix.count() # number of file's lines
#DFngramSix.printSchema()

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     | ngrams|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|[142 3 78 3 4 5, 3 78 3 4 5 221, 78 3 4 5 221 1...|
|[311 331 168 168 42 221, 331 168 168 42 221 221...|
|[6 11 192 6 192 125, 11 192 6 192 125 125, 192 ...|
|[6 54 140 221 11 5, 54 140 221 11 5 6, 140 221 ...|
|[5 63 199 174 5 102, 63 199 174 5 102 102, 199 ...|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

Out[8]: 833

In [9]:
# Algorithm 4 : Count the number of distinct words for a DataFrame DFngramSix of vector of words using Spark
#
#
```



# Agenda

---

- x What Apache Spark is ...
- x Cloud Platform
- x Research Challenges
- x Early results
- x **Future work**
- x References

# Future work

---

- × How can we **speed up** trace analysis with Spark? 
  - ✓ How can we import data (specifically **LTTng traces**) into Spark?
  - × How to filter and structure the data before further processing?
- × How to efficiently trace execution in a Spark environment? 
  - × What are the most useful metrics, analysis and views to develop to study Spark systems?
  - × How can we use **LTTng** to trace Spark?
- × How can we analyse traces in real-time/streaming with Spark?
  - × How can we use the Spark environment to analyse large/"infinite" traces being streamed out of a large system?



# Agenda

---

- × What Apache Spark is ...
- × Cloud Platform
- × Research Challenges
- × Early results
- × Future work
- × **References**

# References

---

- x [1] Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1), 107-113.
- x [2] Karau, H., Konwinski, A., Wendell, P., & Zaharia, M. (2015). *Learning spark: lightning-fast big data analysis*. " O'Reilly Media, Inc."
- x [3] Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark: Cluster computing with working sets. *HotCloud*, 10(10-10), 95.
- x [4] Veiga, J., Expósito, R. R., Pardo, X. C., Taboada, G. L., & Tourifio, J. (2016, December). Performance evaluation of big data frameworks for large-scale data analytics. In *2016 IEEE International Conference on Big Data (Big Data)* (pp. 424-431). IEEE.

# Questions?

---

Thanks for your attention !

Daniel Capelo Borges

daniel.capelo@polymtl.ca