PySpark: DataFrames / SparkSQL + GraphFrames

Norina Akhtar 19643

Table of Contents

Introduction

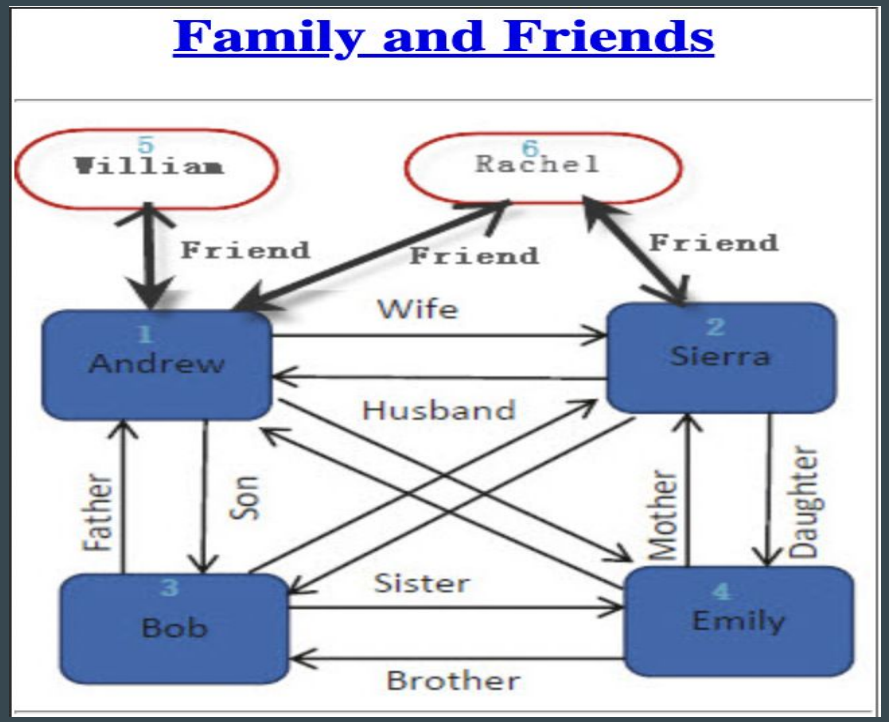
GraphFrames

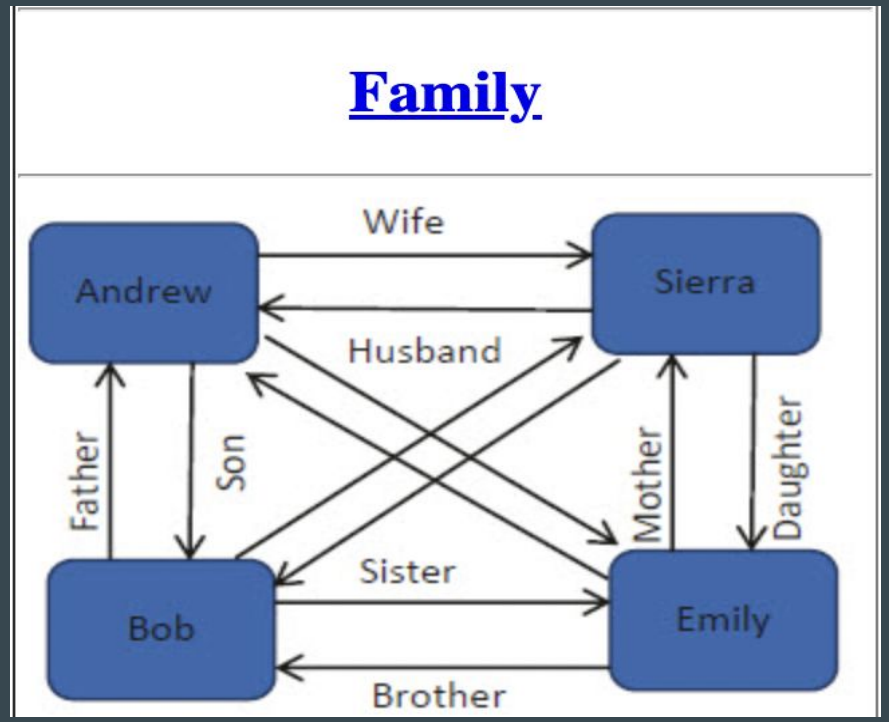
GraphFrames are an abstraction of DataFrames that are used to do Graph Analytics. Graph Analytics stems from the mathematical Graph Theory. Graph Theory is a very important theory used to represent relationships between entities, which we can use to perform various analyses. You are using Graph Theory in your everyday life when using Google.

Google introduced the PageRank algorithm that is based on Graph Theory. It tries to identify the most influential website that suits your search in the best way.

Design

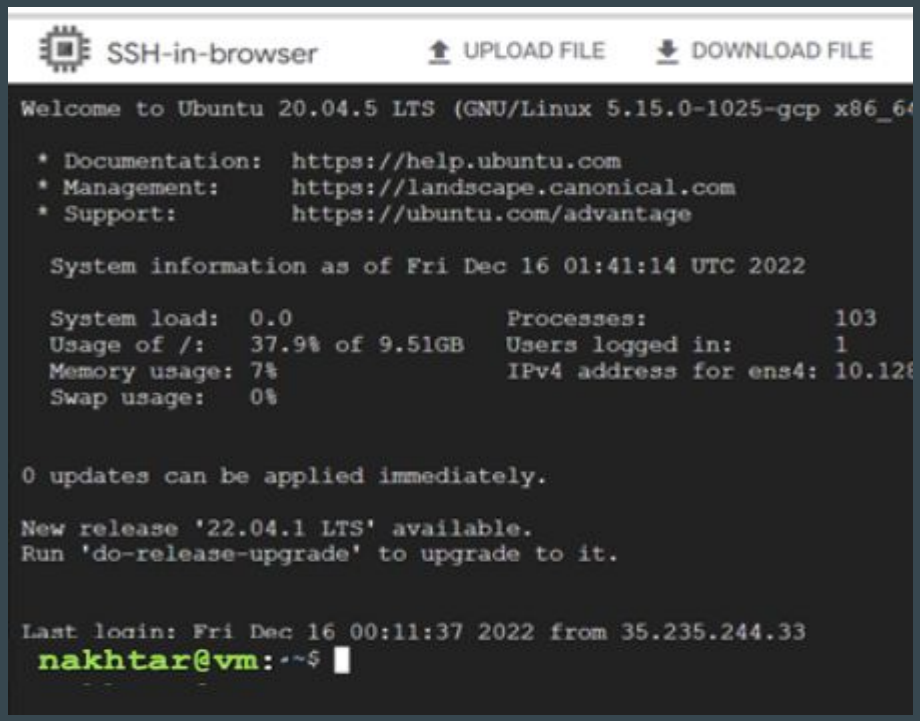
- Source and destination are user ids to relationship column show the relationship between them.





Implementation

1. Create vm instance using compute Engine on GCP and connect to ssh



2. Install pyspark and java

```
nakhtar@vm:~ wget https://archive.apache.org/dist/spark/spark-3.1.3/spark-3.1.3-bin-hadoop2.7.tgz
--2022-12-16 00:14:39--  https://archive.apache.org/dist/spark/spark-3.1.3/spark-3.1.3-bin-hadoop2.7.tgz
Resolving archive.apache.org (archive.apache.org)... 138.201.131.134, 2a01:4f8:172:2ec5::2
Connecting to archive.apache.org (archive.apache.org)|138.201.131.134|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 227452039 (217M) [application/x-gzip]
Saving to: 'spark-3.1.3-bin-hadoop2.7.tgz'

spark-3.1.3-bin-ha 100%[===========>] 216.92M  16.9MB/s    in 14s

2022-12-16 00:14:54 (15.5 MB/s) - 'spark-3.1.3-bin-hadoop2.7.tgz' saved [227452039/227452039]

nakhtar@vm:~$ tar -xvf spark-3.1.3-bin-hadoop2.7.tgz
spark-3.1.3-bin-hadoop2.7/
spark-3.1.3-bin-hadoop2.7/bin/
spark-3.1.3-bin-hadoop2.7/bin/pyspark.cmd
spark-3.1.3-bin-hadoop2.7/bin/spark-submit
spark-3.1.3-bin-hadoop2.7/bin/spark-submit.cmd
spark-3.1.3-bin-hadoop2.7/bin/spark-class2.cmd
spark-3.1.3-bin-hadoop2.7/bin/spark-shell2.cmd
spark-3.1.3-bin-hadoop2.7/bin/pyspark2.cmd
spark-3.1.3-bin-hadoop2.7/bin/docker-image-tool.sh
```

3. Set Environment variable in .bashrc

```
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
export SPARK_HOME=/home/fnagori/spark
export PATH=$SPARKK_HOME/bin:$PATH
export PATH=$SPARK_HOME/sbin:$PATH
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
~
~
~
```

4. Source .bashrc

5. Verify the pyspark

```
nakhtar@vm:~$ source .bashrc
nakhtar@vm:~$ pyspark
Python 3.8.10 (default, Nov 14 2022, 12:59:47)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/home/fnagori/spark-3.1.3-bin-hadoop2.7/jars/spark-unsafe_2.12-3.1.3.jar) to const
ructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
22/12/16 00:43:54 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
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.1.3
      /_/

Using Python version 3.8.10 (default, Nov 14 2022 12:59:47)
Spark context Web UI available at http://instance-1.us-central1-a.c.new-cs570.internal:4040
Spark context available as 'sc' (master = local[*], app id = local-1671151437630).
SparkSession available as 'spark'.
>>>
```

6. Prepare data

```
nakhtar@vm:~$ mkdir in
nakhtar@vm:~$ cd in
nakhtar@vm:~/in$ vi person.csv
nakhtar@vm:~/in$ vi relationship.csv
nakhtar@vm:~/in$ vi pyspark_graphX.py
nakhtar@vm:~/in$
```

person.csv

id	Name	Age
1	Andrew	45
2	Sierra	43
3	Bob	12
4	Emily	10
5	William	35
6	Rachel	32

relation.csv

src	dst	relation
1	2	Husband
1	3	Father
1	4	Father
1	5	Friend
1	6	Friend
2	1	Wife
2	3	Mother
2	4	Mother
2	6	Friend
3	1	Son
3	2	Son
4	1	Daughter
4	2	Daughter
5	1	Friend
6	1	Friend
6	2	Friend

7. Prepare script file as pysprk_graphX.py

```
# Import PySpark
import pyspark
from pyspark.sql import SparkSession

#Create SparkSession
spark = SparkSession.builder.master("local[1]").appName("pysparkGraphX").getOrCreate()

from graphframes import *

# Recipe 9-1. Create GraphFrames
#     person dataframe : id, Name, age
personsDf = spark.read.csv('in/person.csv',header=True, inferSchema=True)

# Create a "persons" SQL table from personsDF DataFrame
personsDf.createOrReplaceTempView("persons")
spark.sql("select * from persons").show()

# relationship dataframe : src, dst, relation
relationshipDf = spark.read.csv('in/relationship.csv',header=True, inferSchema=True)
relationshipDf.createOrReplaceTempView("relationship")
spark.sql("select * from relationship").show()

# - Create a GraphFrame from both person and relationship dataframes
#     >>> graph
#     GraphFrame(v:[id: int, Name: string ... 1 more field], e:[src:
#     int, dst: int ... 1 more field])
# - A GraphFrame that contains v and e.
#   + The v represents vertices and e represents edges.
graph = GraphFrame(personsDf, relationshipDf)
```

```python
# - Degrees represent the number of edges that are connected to a vertex.
#   + GraphFrame supports inDegrees and outDegrees.
#     - inDegrees give you the number of incoming links to a vertex.
#     - outDegrees give the number of outgoing edges from a node.
# - Find all the edges connected to Andrew.
graph.degrees.filter("id = 1").show()

# Find the number of incoming links to Andrew
graph.inDegrees.filter("id = 1").show()

# Find the number of links coming out from Andrew using the outDegrees
graph.outDegrees.filter("id = 1").show()

# Recipe 9-2. Apply Triangle Counting in a GraphFrame
# - Find how many triangle relationships the vertex is participating in
personsTriangleCountDf = graph.triangleCount()
personsTriangleCountDf.show()

# Create a "personsTriangleCount" SQL table from the
# personsTriangleCountDf DataFrame
personsTriangleCountDf.createOrReplaceTempView("personsTriangleCount")

# Create a "personsMaxTriangleCount" SQL table from the
# maxCountDf DataFrame
maxCountDf = spark.sql("select max(count) as max_count from personsTriangleCount")
maxCountDf.createOrReplaceTempView("personsMaxTriangleCount")

spark.sql("select * from personsTriangleCount P JOIN (select * from personsMaxTriangleCount) M ON (M.max_count = P.count) ").show()

# Recipe 9-3. Apply a PageRank Algorithm
pageRank = graph.pageRank(resetProbability=0.20, maxIter=10)
pageRank.vertices.printSchema()

pageRank.vertices.orderBy("pagerank",ascending=False).show()

pageRank.edges.orderBy("weight",ascending=False).show()

# Recipe 9-4. Apply the Breadth First Algorithm
graph.bfs(fromExpr = "Name='Bob'",toExpr = "Name='William'").show()

graph.bfs(fromExpr = "age < 20", toExpr = "name = 'Rachel'").show()
graph.bfs(fromExpr = "age < 20", toExpr = "name = 'Rachel'", edgeFilter = "relation != 'Son'").show()
```

8. Pip3 install numpy

```
nakhtar@vm:~/in$ pip3 install numpy
Collecting numpy
  Downloading numpy-1.23.5-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.1 MB)
     |[redacted]| 17.1 MB 4.8 MB/s
Installing collected packages: numpy
  WARNING: The scripts f2py, f2py3 and f2py3.8 are installed in '/home/fnagori/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed numpy-1.23.5
nakhtar@vm:~/in$
```

9. Submit the job

```
nakhtar@vm:~$ spark-submit --packages graphframes:graphframes:0.8.2-spark3.1-s_2.12 pyspark_graphX.py
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/home/fnagori/spark-3.1.3-bin-hadoop2.7/jars/spark-unsafe_2.12-3.1.3.jar) to const
ructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
:: loading settings :: url = jar:file:/home/fnagori/spark-3.1.3-bin-hadoop2.7/jars/ivy-2.4.0.jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/fnagori/.ivy2/cache
The jars for the packages stored in: /home/fnagori/.ivy2/jars
graphframes#graphframes added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-460dfc75-7f5c-4919-83f8-e4813f157c2c;1.0
        confs: [default]
        found graphframes#graphframes;0.8.2-spark3.1-s_2.12 in spark-packages
        found org.slf4j#slf4j-api;1.7.16 in central
:: resolution report :: resolve 330ms :: artifacts dl 8ms
        :: modules in use:
        graphframes#graphframes;0.8.2-spark3.1-s_2.12 from spark-packages in [default]
        org.slf4j#slf4j-api;1.7.16 from central in [default]
        ---------------------------------------------------------------------
        |                  |            modules            ||   artifacts   |
        |       conf       | number| search|dwnlded|evicted|| number|dwnlded|
        ---------------------------------------------------------------------
        |      default     |   2   |   0   |   0   |   0   ||   2   |   0   |
        ---------------------------------------------------------------------
:: retrieving :: org.apache.spark#spark-submit-parent-460dfc75-7f5c-4919-83f8-e4813f157c2c
        confs: [default]
        0 artifacts copied, 2 already retrieved (0kB/8ms)
22/12/16 01:15:55 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
22/12/16 01:15:56 INFO SparkContext: Running Spark version 3.1.3
22/12/16 01:15:56 INFO ResourceUtils: ==============================================================
22/12/16 01:15:56 INFO ResourceUtils: No custom resources configured for spark.driver.
22/12/16 01:15:56 INFO ResourceUtils: ==============================================================
22/12/16 01:15:56 INFO SparkContext: Submitted application: pysparkGraphX
22/12/16 01:15:56 INFO ResourceProfile: Default ResourceProfile created, executor resources: Map(cores -> name: cores, amount: 1, script: , vendor: , memory ->
 name: memory, amount: 1024, script: , vendor: , offHeap -> name: offHeap, amount: 0, script: , vendor: ), task resources: Map(cpus -> name: cpus, amount: 1.0)
```

Result

GraphFrame

id	Name	Age
1	Andrew	45
2	Sierra	43
3	Bob	12
4	Emily	10
5	William	35
6	Rachel	32

src	dst	relation
1	2	Husband
1	3	Father
1	4	Father
1	5	Friend
1	6	Friend
2	1	Wife
2	3	Mother
2	4	Mother
2	6	Friend
3	1	Son
3	2	Son
4	1	Daughter
4	2	Daughter
5	1	Friend
6	1	Friend
6	2	Friend

Triangle count

id	degree
1	10

id	inDegree
1	5

id	outDegree
1	5

count	id	Name	Age
3	1	Andrew	45
1	6	Rachel	32
1	3	Bob	12
0	5	William	35
1	4	Emily	10
3	2	Sierra	43

count	id	Name	Age	max_count
3	1	Andrew	45	3
3	2	Sierra	43	3

Result

Page Rank

```
root
 |-- id: integer (nullable = true)
 |-- Name: string (nullable = true)
 |-- Age: integer (nullable = true)
 |-- pagerank: double (nullable = true)

+---+--------+---+------------------+
| id|    Name|Age|          pagerank|
+---+--------+---+------------------+
|  1| Andrew| 45| 1.7879231218974 72|
|  2| Sierra| 43| 1.4060167950827 52|
|  6| Rachel| 32|0.7723665979473922|
|  4|  Emily| 10|0.7723665979473922|
|  3|    Bob| 12|0.7723665979473922|
|  5|William| 35|0.4889602891776001|
+---+--------+---+------------------+
```

src	dst	relation	weight
5	1	Friend	1.0
3	1	Son	0.5
4	1	Daughter	0.5
4	2	Daughter	0.5
6	1	Friend	0.5
3	2	Son	0.5
6	2	Friend	0.5
2	3	Mother	0.25
2	4	Mother	0.25
2	1	Wife	0.25
2	6	Friend	0.25
1	2	Husband	0.2
1	6	Friend	0.2
1	3	Father	0.2
1	4	Father	0.2
1	5	Friend	0.2

Result

BFS:

```
+------------+------------+---------------+---------------+-----------------+
|        from|          e0|             v1|             e1|               to|
+------------+------------+---------------+---------------+-----------------+
|{3, Bob, 12}|{3, 1, Son}|{1, Andrew, 45}|{1, 5, Friend}|{5, William, 35}|
+------------+------------+---------------+---------------+-----------------+

+---------------+-----------------+---------------+---------------+----------------+
|           from|               e0|             v1|             e1|              to|
+---------------+-----------------+---------------+---------------+----------------+
|{4, Emily, 10}|{4, 1, Daughter}|{1, Andrew, 45}|{1, 6, Friend}|{6, Rachel, 32}|
|  {3, Bob, 12}|     {3, 1, Son}|{1, Andrew, 45}|{1, 6, Friend}|{6, Rachel, 32}|
|{4, Emily, 10}|{4, 2, Daughter}|{2, Sierra, 43}|{2, 6, Friend}|{6, Rachel, 32}|
|  {3, Bob, 12}|     {3, 2, Son}|{2, Sierra, 43}|{2, 6, Friend}|{6, Rachel, 32}|
+---------------+-----------------+---------------+---------------+----------------+

+---------------+-----------------+---------------+---------------+----------------+
|           from|               e0|             v1|             e1|              to|
+---------------+-----------------+---------------+---------------+----------------+
|{4, Emily, 10}|{4, 1, Daughter}|{1, Andrew, 45}|{1, 6, Friend}|{6, Rachel, 32}|
|{4, Emily, 10}|{4, 2, Daughter}|{2, Sierra, 43}|{2, 6, Friend}|{6, Rachel, 32}|
+---------------+-----------------+---------------+---------------+----------------+
```

Delete the instance

Enhancement Ideas

We can test graphframes using other APIs like java and scala

Saving & loading graphs: GraphFrames fully support Dataframe data source, allowing writing and reading graphs using many formats like Parquet and json.

We can phrase queries using powerful api of spark sql and dataframes

References

https://towardsdatascience.com/graphframes-in-jupyter-a-practical-guide-9b3b346cebc5#:~:text=The%20functionality%20of%20GraphFrames%20and,browsing%20through%20the%20API%20documentation.

https://www.baeldung.com/spark-graph-graphframes

https://hc.labnet.sfbu.edu/~henry/sfbu/course/pyspark_sql_recipes/graphframes/slide/exercise_graphframes.html

Conclusion

Apache Spark is a great tool for computing a relevant amount of data in an optimized and distributed way. And, the GraphFrames library allows us to **easily distribute graph operations over Spark**.