Paul 6 - examples …

```
%pyspark                                                                    FINISHED

# Zeppelin notebook to extract host and in/out-link examples for each of the PLDs in
    the CommonCrawl webgraph
# Complements summaries produced in 'Paul 5', and gets combined with these in 'Paul 7'
    .
# Recomended config for complete run: 4xr4.8xlarge, and set spark.driver.maxResultSize
    to 16g
# PJ - 3 November 2017

import boto
from pyspark.sql.types import *

# Load the saved files from Paul 5.
loadURI="s3://billsdata.net/CommonCrawl/hyperlinkgraph/cc-main-2017-may-jun-jul
    /domaingraph/vertices/"
pld_df_tmp=spark.read.load(loadURI) #.limit(100) #.repartition(256)
```

```
+---+-------+
| ID|    PLD|
+---+-------+
|  0|  aaa.a|
|  1| aaa.aa|
|  2|aaa.aaa|
+---+-------+
only showing top 3 rows
DataFrame[ID: bigint, PLD: string]
```

```
%pyspark                                                                    FINISHED

# Firstly, define a reverse domain function
from pyspark.sql.functions import udf
def reverse_domain(domain):
    return '.'.join(reversed(domain.split('.')))
print(reverse_domain("com.facebook"))
udf_reverse_domain = udf(reverse_domain, StringType())

# Create a reversed version of the PLD dataframe
pld_unrev_df=pld_df.withColumn("PLD_unrev", udf_reverse_domain("PLD")).drop("PLD"
    ).withColumnRenamed("PLD_unrev", "PLD")
```

```
facebook.com
+---+----------+
| ID|       PLD|
+---+----------+
|  0|     a.aaa|
|  1|    aa.aaa|
|  2|   aaa.aaa|
|  3|  aaaa.aaa|
|  4|aaaaaa.aaa|
+---+----------+
only showing top 5 rows
```

```
%pyspark

# Next import the PLD edges as a DataFrame - i.e. in/out links
loadURI="s3://billsdata.net/CommonCrawl/hyperlinkgraph/cc-main-2017-may-jun-jul
    /domaingraph/edges/"
pld_edges_df=spark.read.load(loadURI) #.limit(1000) #.limit(100000000).repartition(8)
    # TODO: Remove temp limit once avoiding spark context shutdown!!
```

FINISHED

```
+---+--------+
|src|     dst|
+---+--------+
|  2| 9193244|
| 20|75600973|
| 21|46356172|
+---+--------+
only showing top 3 rows
DataFrame[src: bigint, dst: bigint]
```

```
%pyspark

# Join dataframes using fast equi-joins (equivalent to reduceByKey on RDDs) - avoids
    the need to create and broadcast an ID-PLD dictionary
pld_edges_df1=pld_unrev_df.join(pld_edges_df, pld_unrev_df.ID==pld_edges_df.src).drop
    ("ID").drop("src").withColumnRenamed("PLD","PLD_src") #rdd.union(rdd1).reduceByKey
    (lambda x,y : x+y)
```

FINISHED

```
+--------------------+--------+
|             PLD_src|     dst|
+--------------------+--------+
|             nic.abb| 9258593|
|             nic.abb|46356172|
|corelaboratory.ab...|      42|
|corelaboratory.ab...|      51|
|corelaboratory.ab...|      53|
+--------------------+--------+
only showing top 5 rows
+--------------------+--------------------+
|            PLD_src|             PLD_dst|
+--------------------+--------------------+
|          icann.org|             nic.abb|
|      wikipedia.org|             nic.abb|
|       namestat.org|             nic.abb|
|    websitelists.in|corelaboratory.ab...|
|hercegovinalijek.ba|corelaboratory.ab...|
```

```
%pyspark

# Load the host-level graph vertices in the same way
saveURI="s3://billsdata.net/CommonCrawl/hyperlinkgraph/cc-main-2017-may-jun-jul
    /hostgraph/vertices/"
host_df=spark.read.load(saveURI) #.repartition(64)
host_df.show(3)
host_df.cache()
```

FINISHED

```
+------+-------+
|hostid|   host|
+------+-------+
|     0|  aaa.a|
|     1| aaa.aa|
|     2|aaa.aaa|
+------+-------+
only showing top 3 rows
DataFrame[hostid: string, host: string]
```

```
%pyspark

# Debug partitioning of our 3 big dataframes
print(pld_df.rdd.getNumPartitions())
print(pld_edges_df.rdd.getNumPartitions())
print(host_df.rdd.getNumPartitions())
```

ERROR

```
192
```

```
Traceback (most recent call last):
  File "/tmp/zeppelin_pyspark-1041518579918016243.py", line 367, in <module>
    raise Exception(traceback.format_exc())
Exception: Traceback (most recent call last):
  File "/tmp/zeppelin_pyspark-1041518579918016243.py", line 355, in <module>
    exec(code, _zcUserQueryNameSpace)
  File "<stdin>", line 2, in <module>
NameError: name 'pld_edges_df' is not defined
```

```
%pyspark                                                                    FINISHED

# DON'T DO THIS - KILLS EXECUTOR MEMORY ON SLAVES, AND DOESN'T SCALE!

# Create a dictionary of PLDs (for ID to PLD mapping of in/out links)
#pld_dict=pld_df.rdd.collectAsMap()
#print(pld_dict[2])

# Distribute and test
#pld_dict_distrib=sc.broadcast(pld_dict) # Maybe broadcasting this huge map to the
    slaves is what's causing them to die!
#print(pld_dict_distrib.value[2]) # Should be aaa.aaa
```

Avoided badness...

```
%pyspark                                                                    FINISHED

# DON'T DO THIS EITHER...

# Function to lookup and unreverse PLDs
#from pyspark.sql.functions import udf
#def reverse_domain_from_ID(id):
#    domain=pld_dict_distrib.value[id]
#    #domain=pld_dict[id] # This will be horribly slow since all queries will have to
    go via the driver but it might work.
#    return '.'.join(reversed(domain.split('.')))
#print(reverse_domain_from_ID(2002))
#udf_reverse_domain_from_ID = udf(reverse_domain_from_ID, StringType())

#  First, create a new edges dataframe consisting of unreversed PLDs
#pld_edges_df2=pld_edges_df.withColumn("src2",udf_reverse_domain_from_ID("src")).drop
    ("src").withColumn("dst2",udf_reverse_domain_from_ID("dst")).drop("dst")
##pld_edges_df.unpersist()
```

Avoided more badness...

```
%pyspark                                                                    FINISHED

# Save this new edges dataframe for future use in parquet format, so we can reload
    later without having the big dict in memory!
#edgesURI="s3://billsdata.net/CommonCrawl/domain_examples_new_edges5/"
```

```
#pld_edges_df2.write.save(edgesURI)
#pld_edges_df2=spark.read.load(edgesURI)
#print(pld_edges_df2.rdd.getNumPartitions())
```

Code to load/save.

```
%pyspark

# Next use reduceByKey to aggregate and ensure no more than 10 per PLD - note we
    create a list for the map values (then + appends)
out_schema = StructType([StructField('PLDout', StringType(), False),StructField
    ('exampleOutLinkPlds', StringType(), False)])
out_degree_examples_df=pld_edges_df2.rdd.map(lambda x:(x['PLD_src'],[x['PLD_dst']]
    )).reduceByKey(lambda acc,pld: acc if len(acc)>=10 else acc+pld).toDF(out_schema)
out_degree_examples_df.show(10)
```

FINISHED

```
+--------------------+--------------------+
|              PLDout|  exampleOutLinkPlds|
+--------------------+--------------------+
|  luttespaysannes.be|[nihil-obstat.be,...|
|        biporteur.fr|[nantescargobike....|
|cooperativadedise...|[vimeo.com, youtu...|
|     atouchofwood.be|[proxi.tools, pro...|
|ergonomie-stockag...|           [ovh.net]|
|       musicordes.ch|      [websaiten.ch]|
|  thevalleylocal.net|[rrpicturearchive...|
|ydk-international.de|[deenet.org, wfg-...|
|   famhillenbrand.eu|             [df.eu]|
|           fowtcg.de|     [fowsystem.com]|
+--------------------+--------------------+
only showing top 10 rows
```

```
%pyspark

# Now do the same for in-degrees
in_schema = StructType([StructField('PLDin', StringType(), False),StructField
    ('exampleInLinkPlds', StringType(), False)])
in_degree_examples_df=pld_edges_df2.rdd.map(lambda x:(x['PLD_dst'],[x['PLD_src']]
    )).reduceByKey(lambda acc,pld: acc if len(acc)>=10 else acc+pld).toDF(in_schema)
```

FINISHED

```
+--------------------+--------------------+
|               PLDin|   exampleInLinkPlds|
+--------------------+--------------------+
|           y-ota.com|          [hama2.jp]|
|ianhenrysimmonds....|[pennedinthemargi...|
|       sexbest24.com|[sharpei-apso.ru,...|
|beachviewflorists...|[list-of-domains....|
|   agent-fashion.com|     [mirnevest.com]|
|        coralclub.ru|[incatalogues.ru,...|
|        komaroku.com|[websitelists.in,...|
|      georgesiga.com|[unbelievable-fac...|
|    eadsummit.com.br|[vidadestartup.or...|
|  notengowebsite.com|[lunadominante.co...|
+--------------------+--------------------+
only showing top 10 rows
DataFrame[PLD_src: string, PLD_dst: string]
```

```
%pyspark                                                                    FINISHED

# Join the In/Out-Link examples together
pld_df_joined=out_degree_examples_df.join(in_degree_examples_df,
    out_degree_examples_df.PLDout==in_degree_examples_df.PLDin, "outer").drop("PLDout"
    )
out_degree_examples_df.unpersist()
in_degree_examples_df.unpersist()
pld_df_joined.show(5)
```

```
+--------------------+--------------------+--------------------+
|  exampleOutLinkPlds|               PLDin|   exampleInLinkPlds|
+--------------------+--------------------+--------------------+
|                null|0----------------...|         [nomina.ru]|
|[wordpress.org, g...|           0-0la.com|[jessicawilson.co...|
|                null|           0-3-6.com|[3d114.com, menok...|
|                null|           0-3ani.ro|[cere.ro, adedir....|
|                null|           0-5-1.com|    [allthecom.info]|
+--------------------+--------------------+--------------------+
only showing top 5 rows
90661578
```

```
%pyspark                                                                    FINISHED

# Save the in-out examples to S3 so we can just load them next time, and avoid the
    above expensive processing!
#pld_df_joined2=pld_df_joined.select("PLDin","exampleInLinkPlds","exampleOutLinkPlds"
    ).withColumnRenamed("PLDin","PLD")
linkExamplesURI="s3://billsdata.net/CommonCrawl/domain_examples_links/"
#pld_df_joined.write.save(linkExamplesURI)
```

```
200
```

```
%pyspark                                                                    FINISHED

# Next, we'll construct a local dictionary from of all the PLDS (key is the PLD, value
    is the ID)
# This is our truth-table of known PLDs that we'll use when extracting host examples

# Create a bloom filter using a pure python package (might be a little slow)
from pybloom import BloomFilter
pld_bf = BloomFilter(capacity=91000000, error_rate=0.005)

for row in pld_df.rdd.collect(): #.take(10000): # limit(10000000) # TODO: Still bad
    (and exceeds spark.driver.maxResultSize with all rows)!
    pld_bf.add(row['PLD'])

#print(pld_df.rdd.take(3))
#print(pld_df.rdd.take(3)[2]['PLD'])
print("aaa.aaa" in pld_bf) # Should be True

import sys
print(sys.getsizeof(pld_bf))
print(len(pld_bf)) # Should match number of items entered

# Broadcast the bloom filter so it's available on all the slave nodes - we don't need
    to change
# it any more so it's fine being immutable.
pld_bf_distrib=sc.broadcast(pld_bf)

print("aaa.aaa" in pld_bf) # Should be true
```

```
True
64
90751305
True
False
True
False
```

```
%pyspark                                                                    FINISHED

# Returns a Boolean to say whether PLD is a hostname in itself
def is_a_pld(hostname):
    #if hostname in pld_lookup_table:
    #if pld_lookup_table.filter(lambda a: a == hostname).count()>0:
    if hostname in pld_bf_distrib.value:
        return True
    else:
        return False

# Function to do the hostname->pld conversion, if the reversed pld exists in our
    dictionary
def convert_hostname(hostname):
    # Return hostname as-is, if this is already a PLD
    #if hostname in pld_lookup_table:
    #if pld_lookup_table.filter(lambda a: a == hostname).count()>0:
```

```
    if hostname in pld_bf_distrib.value:
        return hostname
    # Otherwise we're going to have to split it up and test the parts
    try:
        parts=hostname.split('.')
        if (len(parts)>4 and is_a_pld('.'.join(parts[0:4]))):
            return '.'.join(parts[0:4])
        if (len(parts)>3 and is_a_pld('.'.join(parts[0:3]))):
            return '.'.join(parts[0:3])
        if (len(parts)>2 and is_a_pld('.'.join(parts[0:2]))):
            return '.'.join(parts[0:2])
        if (len(parts)>1 and is_a_pld('.'.join(parts[0:1]))):
            return '.'.join(parts[0:1])
        return "ERROR" # Couldn't find a corresponding PLD - this should never happen!
    except:
        return "ERROR"

# Test
print(convert_hostname("aaa.aaa"))
```

```
aaa.aaa
True
```

```
%pyspark                                                                FINISHED

# Generate 10 host examples per PLD.

# Firstly, define a reverse domain function
def reverse_domain(domain):
    return '.'.join(reversed(domain.split('.')))
print(reverse_domain("com.facebook"))
#udf_reverse_domain = udf(reverse_domain, StringType())

# Now reverse all host names after conversion to PLDs (including lookup) but prior to
    summarization.
#host_example_rdd=unrev_host_df.rdd.map(lambda x: (convert_hostname(x['host']
    ),[x['host']])).reduceByKey(lambda acc,host: acc if len(acc)>=10 else acc+host)
host_example_rdd=host_df.rdd.map(lambda x: (reverse_domain(convert_hostname(x['host']
    )),[reverse_domain(x['host'])])).reduceByKey(lambda acc,host: acc if len(acc)>=10
    else acc+host)
```

```
facebook.com
[(u'eddyseel.com', [u'eddyseel.com']), (u'qkglxvbiyphb.com', [u'qkglxvbiyphb.com']), (u'
sexbest24.com', [u'sexbest24.com']), (u'cmtcdmrbyzbd.com', [u'cmtcdmrbyzbd.com']), (u'gf
xthai.org', [u'gfxthai.org']), (u'sieheunten.de', [u'sieheunten.de']), (u'komaroku.com',
[u'komaroku.com', u'novel.komaroku.com', u'test.komaroku.com']), (u'kerbalaya.net', [u'k
erbalaya.net']), (u'monclerjacketssales2012.com', [u'monclerjacketssales2012.com']), (u'
restaurantdurmitor.com', [u'restaurantdurmitor.com']), (u'violetaorgaz.com', [u'violetao
rgaz.com']), (u'fa-altmark.de', [u'fa-altmark.de']), (u'agapelive.co.za', [u'agapelive.c
o.za']), (u'fccv.org.ve', [u'fccv.org.ve']), (u'vitapharmed.org', [u'vitapharmed.org']),
(u'truckandplantonline.com', [u'truckandplantonline.com']), (u'techbrosnetworks.net', [u
'techbrosnetworks.net']), (u'libre-planete.fr', [u'libre-planete.fr']), (u'blackhattersg
uide.com', [u'blackhattersguide.com']), (u'vibeffe.it', [u'vibeffe.it'])]
```

```
%pyspark

#print(host_example_rdd.take(100))

# Convert host examples back to a dataframe
out_schema = StructType([StructField('PLD', StringType(), False),StructField
    ('exampleHosts', StringType(), False)])
host_examples_df=host_example_rdd.toDF(out_schema)
```

FINISHED

```
+--------------------+--------------------+
|                 PLD|        exampleHosts|
+--------------------+--------------------+
|  galeriaccrj.com.br|[galeriaccrj.com.br]|
|cooperativadedise...|[cooperativadedis...|
|klemens-transport...|[klemens-transpor...|
|   agent-fashion.com| [agent-fashion.com]|
| xishannongjiale.com|[xishannongjiale....|
|         gfxthai.org|       [gfxthai.org]|
|       sieheunten.de|     [sieheunten.de]|
|       carvenrose.gr|     [carvenrose.gr]|
|      georgesiga.com|    [georgesiga.com]|
|restaurantdurmito...|[restaurantdurmit...|
| zhangxiaohui.com.cn|[zhangxiaohui.com...|
|       fa-altmark.de|     [fa-altmark.de]|
|        remibaby.com|      [remibaby.com]|
|         cclb.com.my|       [cclb.com.my]|
|truckandplantonli...|[truckandplanton...|
```

```
%pyspark

# Join in/out-link summaries with host examples dataframe
example_df=pld_df_joined.join(host_examples_df, pld_df_joined.PLDin==host_examples_df
    .PLD, "outer").drop("PLDin").select("PLD","exampleHosts","exampleInLinkPlds"
    ,"exampleOutLinkPlds")
example_df.show(100)
```

FINISHED

```
+--------------------+--------------------+--------------------+--------------------+
|                 PLD|        exampleHosts|   exampleInLinkPlds|  exampleOutLinkPlds|
+--------------------+--------------------+--------------------+--------------------+
|0----------------...|[0---------------...|         [nomina.ru]|                null|
|           0-0la.com|         [0-0la.com]|[jessicawilson.co...|[wordpress.org, g...|
|           0-3-6.com|         [0-3-6.com]|[3d114.com, menok...|                null|
|           0-3ani.ro|         [0-3ani.ro]|[cere.ro, adedir....|                null|
|           0-5-1.com|         [0-5-1.com]|    [allthecom.info]|                null|
|       0-60times.net|     [0-60times.net]|[lodekka.com, kev...|                null|
|            0-744.cn|          [0-744.cn]|     [wordpress.com]|                null|
|0-ads-free-web-pa...|[0-ads-free-web-p...|[list-of-domains....|                null|
|       0-artlove.net|     [0-artlove.net]|[list-of-domains....|                null|
|  0-clubpenguin-0.tk|[0-clubpenguin-0.tk]|  [similarsites.com]|                null|
+--------------------+--------------------+--------------------+--------------------+
only showing top 10 rows
99405158
```

```
%pyspark                                                                    FINISHED

# Save final table to S3 in parquet format, broken into smaller files (for fast
    reading into Paul 7 that will combine original summaries with examples)
outputURI="s3://billsdata.net/CommonCrawl/domain_examples4/"
#codec="org.apache.hadoop.io.compress.GzipCodec"
#example_df.write.format('com.databricks.spark.csv').options(header='true', codec
```

```
%pyspark                                                                    READY

```