Hands On: Building a Degree Histogram Reading | Coursera

Hands On: Building A Degree Histogram

1. Count the number of vertices and edges
2. Define a min and max function for Spark’s reduce method
3. Compute min and max degrees
4. Compute the histogram data of the degree of connectedness

Count the number of vertices and edges

Print the number of links.

input:

```
metrosGraph.numEdges
```

output:

```
res13: Long = 65
```

Print the number of nodes.

input:

```
metrosGraph.numVertices
```

output:

```
res14: Long = 93
```

Define a min and max function for Spark’s reduce method

Define a min and max function.

input:

```
def max(a: (VertexId, Int), b: (VertexId, Int)): (VertexId, Int) = {
  if (a._2 > b._2) a else b
}
```

output:

```
max: (a: (org.apache.spark.graphx.VertexId, Int), b:
(org.apache.spark.graphx.VertexId, Int))(org.apache.spark.graphx.VertexId,
Int)
```

input:

```
def min(a: (VertexId, Int), b: (VertexId, Int)): (VertexId, Int) = {
  if (a._2 <= b._2) a else b
}
```

output:

```
min: (a: (org.apache.spark.graphx.VertexId, Int), b:
(org.apache.spark.graphx.VertexId, Int))(org.apache.spark.graphx.VertexId,
Int)
```

Compute min and max degrees

Find which which VertexId and the edge count of the vertex with the most out edges. (This can be any vertex because all vertices have one out edge.)

input:

```
metrosGraph.outDegrees.reduce(max)
```

output:

```
res15: (org.apache.spark.graphx.VertexId, Int) = (5,1)
```

Print the returned vertex.

input:

```
metrosGraph.vertices.filter(_._1 == 5).collect()
```

output:

```
res16: Array[(org.apache.spark.graphx.VertexId, PlaceNode)] =
Array((5,Metro(Delhi,21753486)))
```

Find which which VertexId and the edge count of the vertex with the most in edges.

input:

```
metrosGraph.inDegrees.reduce(max)
```

output:

```
res17: (org.apache.spark.graphx.VertexId, Int) = (108,14)
```

Print the returned vertex.

input:

```
metrosGraph.vertices.filter(_._1 == 108).collect()
```

output:

```
res18: Array[(org.apache.spark.graphx.VertexId, PlaceNode)] =
Array((108,Country(United States)))
```

Find the number vertexes that have only one out edge.

input:

```
metrosGraph.outDegrees.filter(_._2 <= 1).count
```

output:

```
res19: Long = 65
```

Find the maximum and minimum degrees of the connections in the network.

input:

```
metrosGraph.degrees.reduce(max)
```

output:

```
res20: (org.apache.spark.graphx.VertexId, Int) = (108,14)
```

input:

```
metrosGraph.degrees.reduce(min)
```

output:

```
res21: (org.apache.spark.graphx.VertexId, Int) = (34,1)
```

Compute the histogram data of the degree of connectedness

Print the histogram data of the degrees for countries only.

input:

```
metrosGraph.degrees.
  filter { case (vid, count) => vid >= 100 }. // Apply filter so only 
VertexId < 100 (countries) are included
  map(t => (t._2,t._1)).
  groupByKey.map(t => (t._1,t._2.size)).
  sortBy(_._1).collect()
```

output:

```
res22: Array[(Int, Int)] = Array((1,18), (2,4), (3,2), (5,2), (9,1), (14,1))
```