BigData 2025: Project 3

Project Big Data is provided by University of Tartu.

Students: Kalju Jake Nekvasil, Joosep Orasmäe, Tanel Tiisler, Kaupo Humal

Queries

Query 0: Graph Construction & Initialization

We setup a Spark environment with Data Lake and construct a graph representation of flight data using GraphFrame. We then read and initialize two graph structures from the flight csv data to be used in later queries:

1. `edges`: Unique flight paths
2. `full_edges`: All flight records

Within the graph, vertices represent airports and flight paths represent edges. nodes and all flight paths as edges. These cache the edge and vertex data for improved performance.

Query 1: Graph Metrics

Four network metrics are computed for both the `edges` (for connectivity) and `full_edges` (for intensity) structures:

1. In-degree
2. Out-degree
3. Total degree

Triangle count (`triangle_counts`) is then calculated by creating an undirected version of the graph by adding reverse edges, then identifies common neighbors between connected airports. The triangle count for each airport is made by counting how many times triangular routes occur (divided by 2 for all double edges.)

We output the (top 20 rows of the) final table of metrics for both `edges` and `full_edges`.

Airport	In-Degree (Structure)	Out-Degree (Structure)	Total Degree (Structure)	Triangles	In-Degree (Traffic)	Out-Degree (Traffic)	Total Degree (Traffic)
ABQ	32	31	63	311	35,577	35,582	71,159
ACK	2	1	3	1	343	342	685
ALO	1	1	2	0	331	330	661
ANC	28	27	55	122	17,788	17,791	35,579
AEX	4	4	8	6	2,948	2,947	5,895
AKN	1	1	2	0	77	77	154

Airport	In-Degree (Structure)	Out-Degree (Structure)	Total Degree (Structure)	Triangles	In-Degree (Traffic)	Out-Degree (Traffic)	Total Degree (Traffic)
AUS	36	37	73	416	41,846	41,843	83,689
ABY	1	1	2	0	997	995	1,992
ACV	5	4	9	6	3,364	3,370	6,734
ADK	1	1	2	0	103	103	206
ABE	8	7	15	27	4,037	4,034	8,071
ACY	2	2	4	1	522	522	1,044
AGS	3	3	6	5	3,106	3,107	6,213
ATW	7	7	14	21	5,306	5,303	10,609
ADQ	1	1	2	0	631	631	1,262
AMA	6	6	12	12	6,649	6,649	13,298
ATL	165	165	330	1,761	417,457	417,449	834,906
ABI	1	1	2	0	2,490	2,490	4,980
ACT	1	1	2	0	1,052	1,053	2,105
ALB	18	18	36	142	12,020	12,018	24,038

Query 2: Total Triangle Count

To find the total number of unique triangular routes, we use the triangle counts structure calculated in Query 1. All the counts are summed then divided by three. `triangle_counts` counted each airport once in the cycle, so by dividing by three we simplify from total vertices to total triangles.

total num. of triangles
16015.0

Query 3: Centrality Measure (Eigenvector Centrality)

The chosen centrality measure is Eigenvector Centrality. It measures a node's importance based on how well-connected its neighbors are. Unlike PageRank, it does not use a damping factor and treats all influence as coming directly from connected nodes without accounting for jumps or teleportation from further nodes. In the context of flights, it allows to measure centrality based on direct flights from popular airports.

First, each airport gets assigned an initial score of 1.0. The score is iteratively updated by having nodes inherit influence from their connected neighbors. Airports receive final scores equal to the sum of their neighbors' scores after 10 iterations. Normalization is applied at each iteration by dividing all scores by their total sum.

Results are validated against `NetworkX`'s eigenvector centrality. We see identical results, confirming the correctness of our implementation.

We output the top 20 (sorted by ranking) results of the eigenvector centrality implementation and the validation results.

id	score	q3_rank	nx_score	nx_rank
ATL	0.018841687327681	1	0.018844990076174	1
ORD	0.018110084675545	2	0.018113072392038	2
DFW	0.017016256953545	3	0.017018795580919	3
DTW	0.016770254346926	4	0.016772806665139	4
DEN	0.016185588480573	5	0.016187465560831	5
MSP	0.016167495214740	6	0.016169655541720	6
IAH	0.015682789267655	7	0.015684759450529	7
CVG	0.015123160359758	8	0.015125031640433	8
LAS	0.014596222840050	9	0.014597356212399	9
EWR	0.014355588217040	10	0.014357293204337	10
PHX	0.014328265155341	11	0.014329322069793	11
MEM	0.014090583475359	12	0.014092166585833	12
MCO	0.013613478113955	13	0.013614527854571	13
CLT	0.013598602138314	14	0.013599828601782	14
LAX	0.013278321168654	15	0.013279089986496	15
SLC	0.013227605218655	16	0.013228624072237	16
BWI	0.013205945160438	17	0.013206802915325	17
IAD	0.013022968172385	18	0.013023926697623	18
CLE	0.012996524760205	19	0.012997300879488	19
JFK	0.012822879580062	20	0.012823725245535	20

Query 4: PageRank Algorithm

The PageRank Algorithm is implemented using raw GraphFrames. First, all vertices are initialized with a score of 1. Similarly to Query 3, we complete 10 iterations. For each iteration we calculate how many each vertex contributes to its neighbors and attach that score to the edges. Then create new scores by calculating the incoming contributions for each node and applying damping (0.85). Finally the vertices are updated with the new scores.

The results of our custom PageRank implementation are validated with the built-in PageRank algorithm results. The values of each PageRank implementation are virtually identical, confirming that our implementation is correct.

id	custom_score	built_in_score
ATL	11.335268868250697	11.335268868250694
DFW	8.987200141119203	8.987200141119201
ORD	8.330832084011150	8.330832084011147

id	custom_score	built_in_score
DTW	7.722346489398372	7.722346489398372
MSP	7.592985738022308	7.592985738022308
DEN	7.026287164606544	7.026287164606543
SLC	6.764494781487534	6.764494781487533
IAH	5.662321605756484	5.662321605756484
LAX	5.070949815830091	5.070949815830091
CVG	4.856791376084717	4.856791376084717

The results show that the most "important" airport is the Hartsfield–Jackson Atlanta International Airport (ATL). This also matches with the fact that Atlanta is known to be the busiest airport in the US by quite a margin. All of the other airports in the top 10 are also considered to be very busy but the exact order differs slightly. For example, the Los Angeles International Airport (LAX) is considered to be the second busiest but it is ranked 9th by PageRank. This is likely because the edges in our graph are not weighted in any way (could use something like nr. of passengers per year).

Query 5: Group w/ Most Connected Airports

We manually iterate a maximum of 10 times and vertices (airports) are grouped by the smallest component IDs they can reach. The results show that the entire graph seems to be strongly connected (all airport vertices have an edge to every other airport).

scc_id	airports_in_scc
ABE	[ABQ, ACK, ALO, ...]

num of groups
1

scc_id	num_airports
ABE	296

We validated our results with a built-in methods to confirm the correctness of our search and the strongly connected behavior of the graph. We see this confirmation in the table below.

component	count
0	296