In [2]:

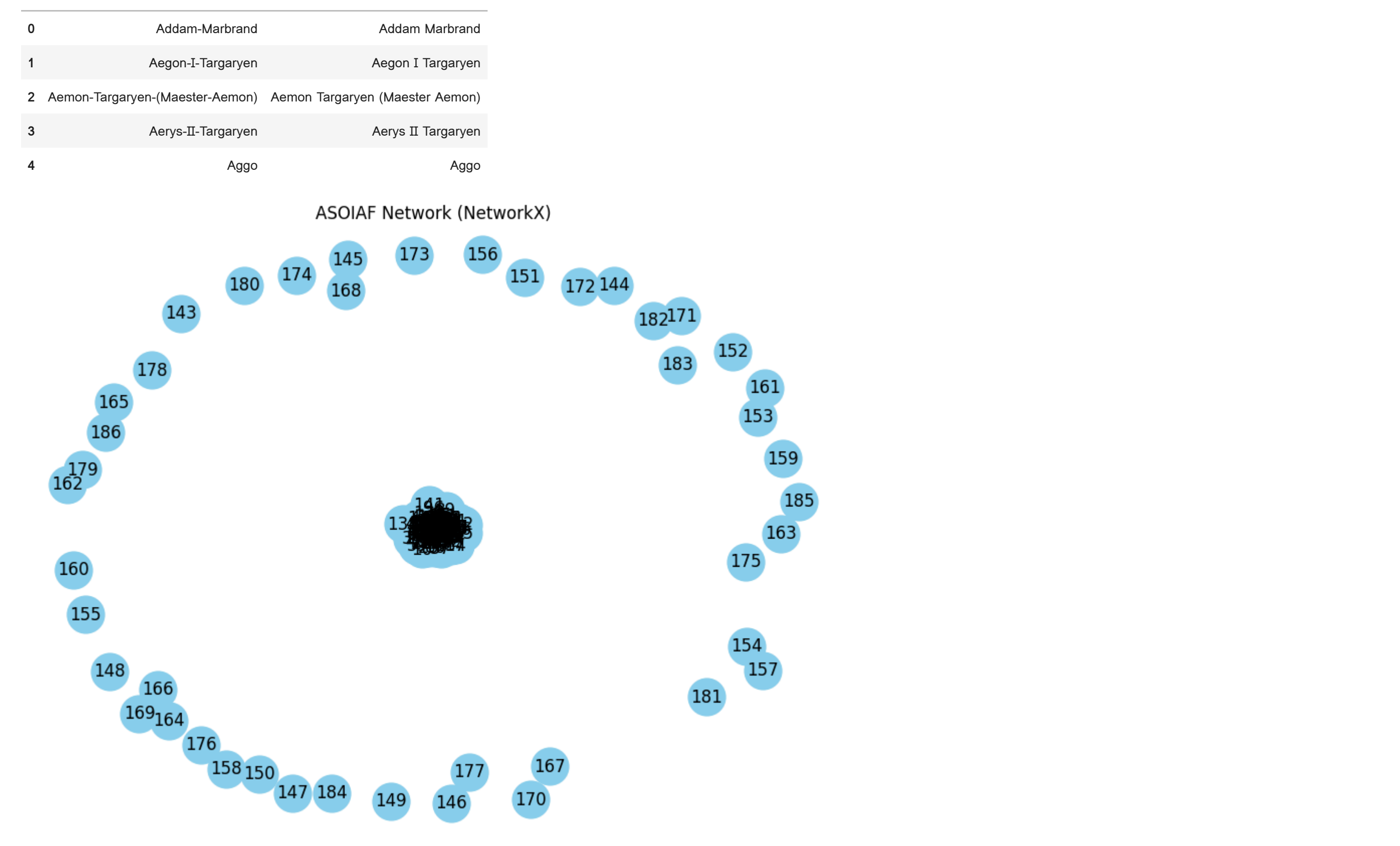
```python
# ASOIAF Graph Analysis

## Imports and Data Loading
import os
import networkx as nx
import matplotlib.pyplot as plt
import igraph as ig
import pandas as pd
from py2neo import Graph

# Define helper functions
def plot_networkx_graph(graph, title):
    plt.figure(figsize=(8, 6))
    nx.draw(graph, with_labels=True, node_color='skyblue', node_size=700, edge_color='gray')
    plt.title(title)
    plt.show()

def plot_centrality_measures(measure, title):
    plt.figure(figsize=(10, 6))
    nodes = list(measure.keys())
    values = list(measure.values())
    plt.bar(nodes, values, color='skyblue')
    plt.xlabel('Nodes')
    plt.ylabel('Centrality Value')
    plt.title(title)
    plt.legend(['Centrality Measure'])
    plt.show()

def plot_igraph_graph(graph, layout, title):
    try:
        ig.plot(graph, layout=layout, vertex_size=20, vertex_label_size=10, vertex_color='skyblue', title=title)
    except ImportError as e:
        print(f"Error plotting igraph: {e}")

# Load Data
edges = pd.read_csv('data/asoiaf-book1-edges.csv')
nodes = pd.read_csv('data/asoiaf-book1-nodes.csv')

# Display DataFrames
print("Edges DataFrame:")
display(edges.head())

print("Nodes DataFrame:")
display(nodes.head())

# Create Graph
nodes['Id'] = nodes['Id'].astype('category').cat.codes
edges['Source'] = edges['Source'].astype('category').cat.codes
edges['Target'] = edges['Target'].astype('category').cat.codes

G_nx = nx.Graph()
for index, row in nodes.iterrows():
    G_nx.add_node(row['Id'], label=row['Label'])

for index, row in edges.iterrows():
    G_nx.add_edge(row['Source'], row['Target'], weight=row['weight'])

# Plot NetworkX Graph
plot_networkx_graph(G_nx, "ASOIAF Network (NetworkX)")

# Create and Plot igraph Graph
G_ig = ig.Graph.DataFrame(edges[['Source', 'Target', 'weight']], directed=False, vertices=nodes[['Id', 'Label']])
layout = G_ig.layout("kk")
plot_igraph_graph(G_ig, layout, "ASOIAF Network (igraph)")

# Perform and Plot Centrality Measures
pagerank_nx = nx.pagerank(G_nx)
print("PageRank (NetworkX):")
pagerank_df = pd.DataFrame(list(pagerank_nx.items()), columns=['Node', 'PageRank'])
display(pagerank_df.sort_values(by='PageRank', ascending=False).head(10))
plot_centrality_measures(pagerank_nx, "PageRank (NetworkX)")

betweenness_nx = nx.betweenness_centrality(G_nx)
print("Betweenness Centrality (NetworkX):")
betweenness_df = pd.DataFrame(list(betweenness_nx.items()), columns=['Node', 'Betweenness'])
display(betweenness_df.sort_values(by='Betweenness', ascending=False).head(10))
plot_centrality_measures(betweenness_nx, "Betweenness Centrality (NetworkX)")

closeness_nx = nx.closeness_centrality(G_nx)
print("Closeness Centrality (NetworkX):")
closeness_df = pd.DataFrame(list(closeness_nx.items()), columns=['Node', 'Closeness'])
display(closeness_df.sort_values(by='Closeness', ascending=False).head(10))
plot_centrality_measures(closeness_nx, "Closeness Centrality (NetworkX)")

degree_nx = nx.degree_centrality(G_nx)
print("Degree Centrality (NetworkX):")
degree_df = pd.DataFrame(list(degree_nx.items()), columns=['Node', 'Degree'])
display(degree_df.sort_values(by='Degree', ascending=False).head(10))
plot_centrality_measures(degree_nx, "Degree Centrality (NetworkX)")

pagerank_ig = G_ig.pagerank()
print("PageRank (igraph):")
pagerank_ig_df = pd.DataFrame(list(enumerate(pagerank_ig)), columns=['Node', 'PageRank'])
display(pagerank_ig_df.sort_values(by='PageRank', ascending=False).head(10))
plot_centrality_measures(dict(enumerate(pagerank_ig)), "PageRank (igraph)")

# Plot Florentine Families Network
G_florentine = nx.florentine_families_graph()
plot_networkx_graph(G_florentine, "Florentine Families Network")

pagerank_florentine = nx.pagerank(G_florentine)
print("Florentine Families PageRank:")
pagerank_florentine_df = pd.DataFrame(list(pagerank_florentine.items()), columns=['Node', 'PageRank'])
display(pagerank_florentine_df.sort_values(by='PageRank', ascending=False).head(10))
plot_centrality_measures(pagerank_florentine, "Florentine Families PageRank")

# Neo4j Integration (if available)
try:
    graph = Graph("bolt://localhost:7687", auth=("neo4j", "Sisma123"))
    graph.delete_all()

    for index, row in nodes.iterrows():
        graph.run("CREATE (n:Character {id: $id, label: $label})", id=row['Id'], label=row['Label'])

    for index, row in edges.iterrows():
        graph.run("""
        MATCH (a:Character {id: $source}), (b:Character {id: $target})
        CREATE (a)-[:INTERACTS {weight: $weight}]->(b)
        """, source=row['Source'], target=row['Target'], weight=row['weight'])

    pagerank_query = """
        CALL gds.pageRank.stream('characterGraph', {
        dampingFactor: 0.85,
        maxIterations: 20
        })
        YIELD nodeId, score
        RETURN gds.util.asNode(nodeId).label AS label, score
        ORDER BY score DESC
    """

    pagerank_results = graph.run(pagerank_query).data()
    print("PageRank (Neo4j):")
    pagerank_neo4j_df = pd.DataFrame(pagerank_results)
    display(pagerank_neo4j_df.head(10))

    query = """
    MATCH (n:Character)-[r:INTERACTS]->(m:Character)
    RETURN n, r, m
    """
    nodes_dict = {}
    G_neo4j = nx.DiGraph()

    for record in graph.run(query):
        n = record['n']
        m = record['m']
        r = record['r']

        if n['id'] not in nodes_dict:
            nodes_dict[n['id']] = n
            G_neo4j.add_node(n['id'], label=n['label'])
        if m['id'] not in nodes_dict:
            nodes_dict[m['id']] = m
            G_neo4j.add_node(m['id'], label=m['label'])

        G_neo4j.add_edge(n['id'], m['id'], weight=r['weight'])

    pagerank_neo4j = nx.pagerank(G_neo4j)
    print("PageRank (Neo4j):")
    pagerank_neo4j_df = pd.DataFrame(list(pagerank_neo4j.items()), columns=['Node', 'PageRank'])
    display(pagerank_neo4j_df.sort_values(by='PageRank', ascending=False).head(10))
    plot_centrality_measures(pagerank_neo4j, "PageRank (Neo4j)")

except Exception as e:
    print(f"Error integrating with Neo4j: {e}")
```

Edges DataFrame:

	Source	Target	Type	weight	book
0	Addam-Marbrand	Jaime-Lannister	Undirected	3	1
1	Addam-Marbrand	Tywin-Lannister	Undirected	6	1
2	Aegon-I-Targaryen	Daenerys-Targaryen	Undirected	5	1
3	Aegon-I-Targaryen	Eddard-Stark	Undirected	4	1
4	Aemon-Targaryen-(Maester-Aemon)	Alliser-Thorne	Undirected	4	1

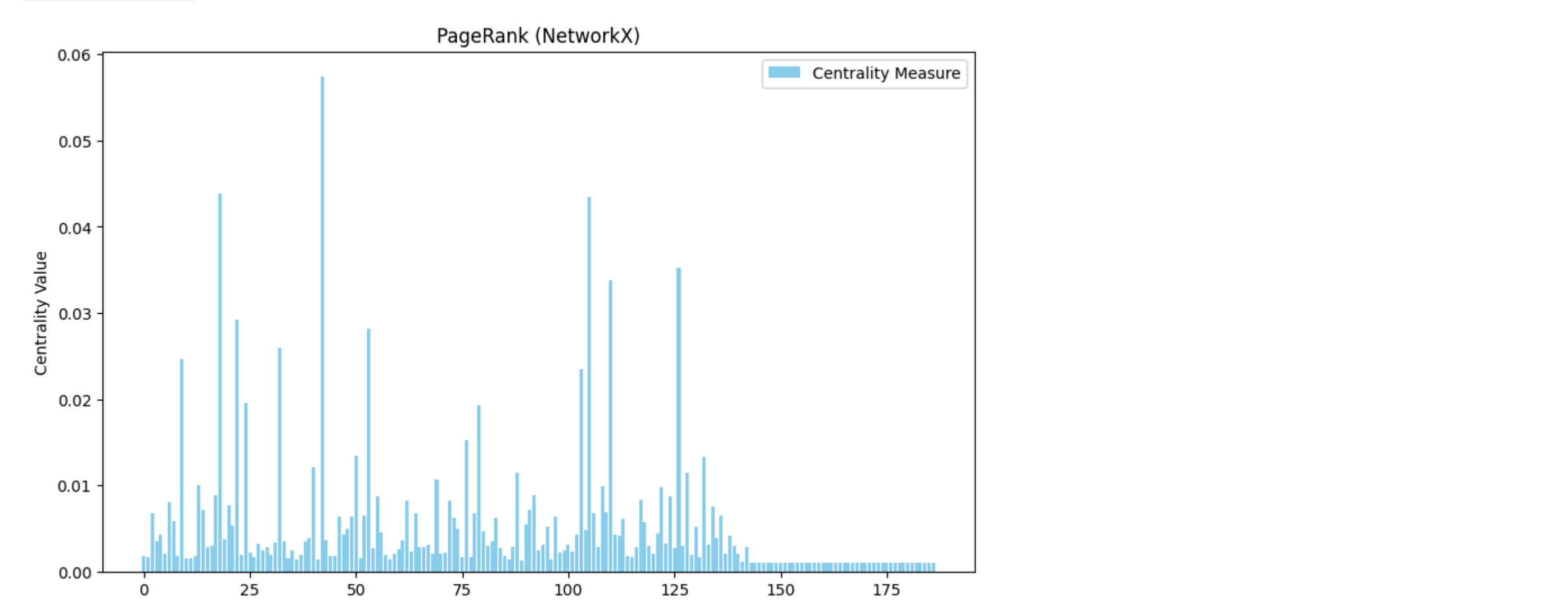
Nodes DataFrame:

	Id	Label
0	Addam-Marbrand	Addam Marbrand
1	Aegon-I-Targaryen	Aegon I Targaryen
2	Aemon-Targaryen-(Maester-Aemon)	Aemon Targaryen (Maester Aemon)
3	Aerys-II-Targaryen	Aerys II Targaryen
4	Aggo	Aggo



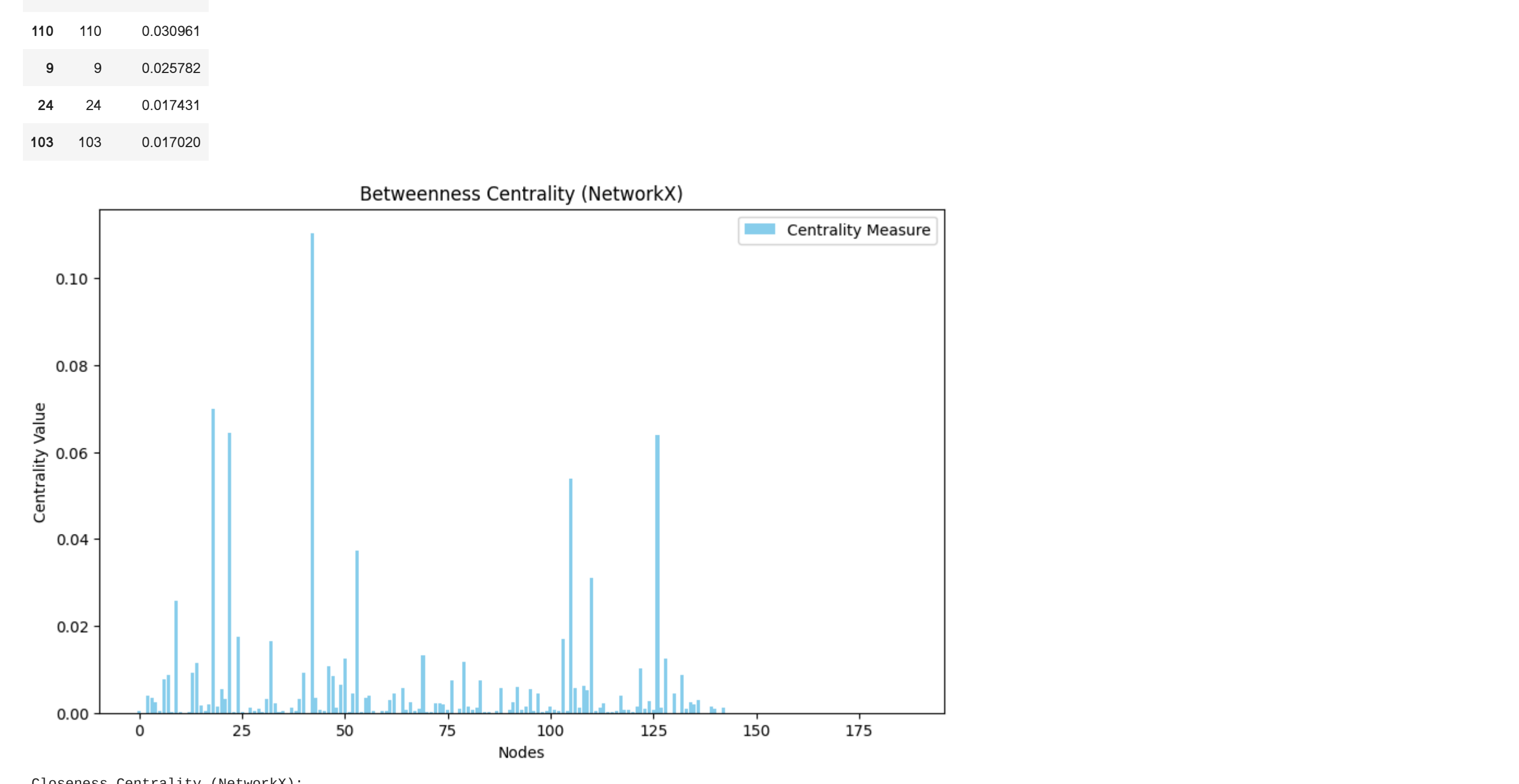
PageRank (NetworkX):

	Node	PageRank
42	42	0.057470
18	18	0.043823
105	105	0.043499
126	126	0.035265
110	110	0.033803
22	22	0.029169
53	53	0.028125
32	32	0.025904
9	9	0.024682
103	103	0.023462



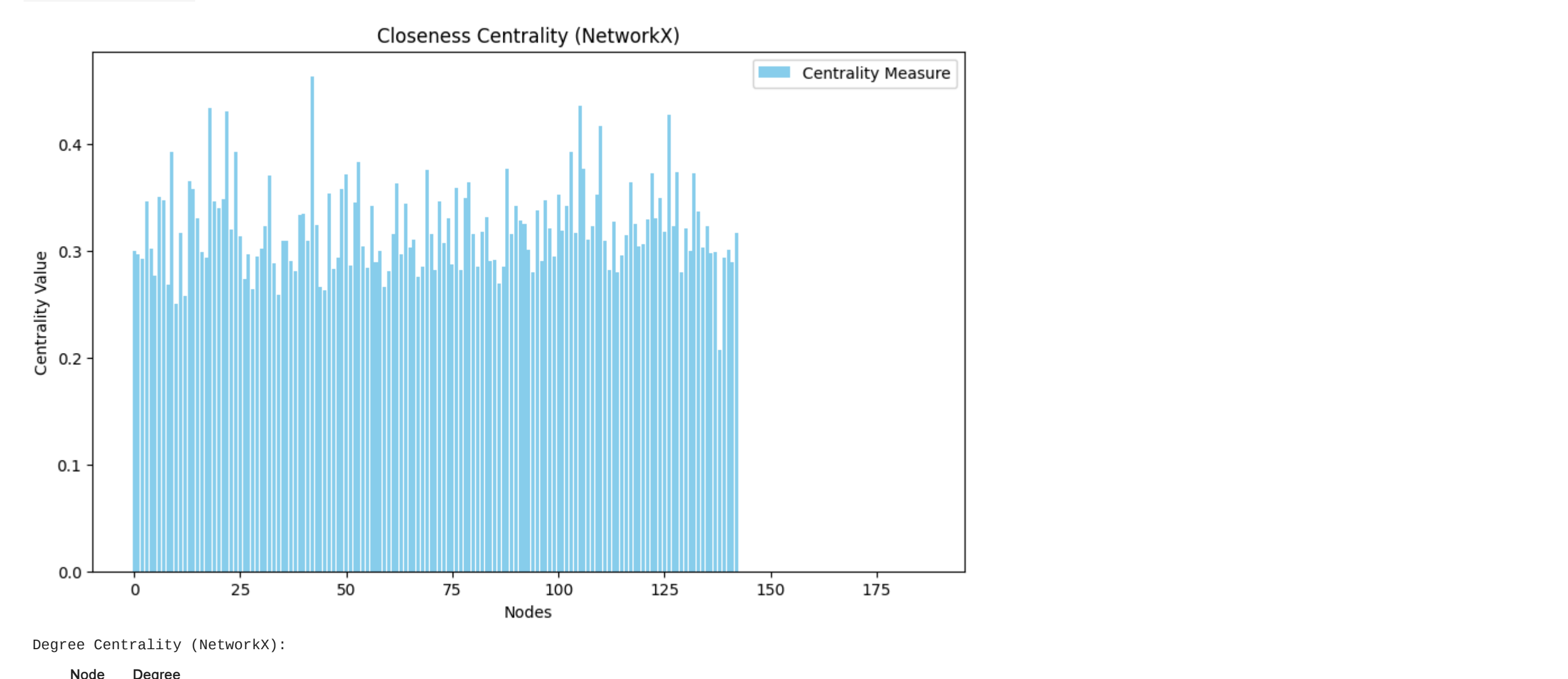
Betweenness Centrality (NetworkX):

	Node	Betweenness
42	42	0.110435
18	18	0.070059
22	22	0.064514
126	126	0.063947
105	105	0.053983
53	53	0.037401
110	110	0.030961
9	9	0.025782
24	24	0.017431
103	103	0.017020



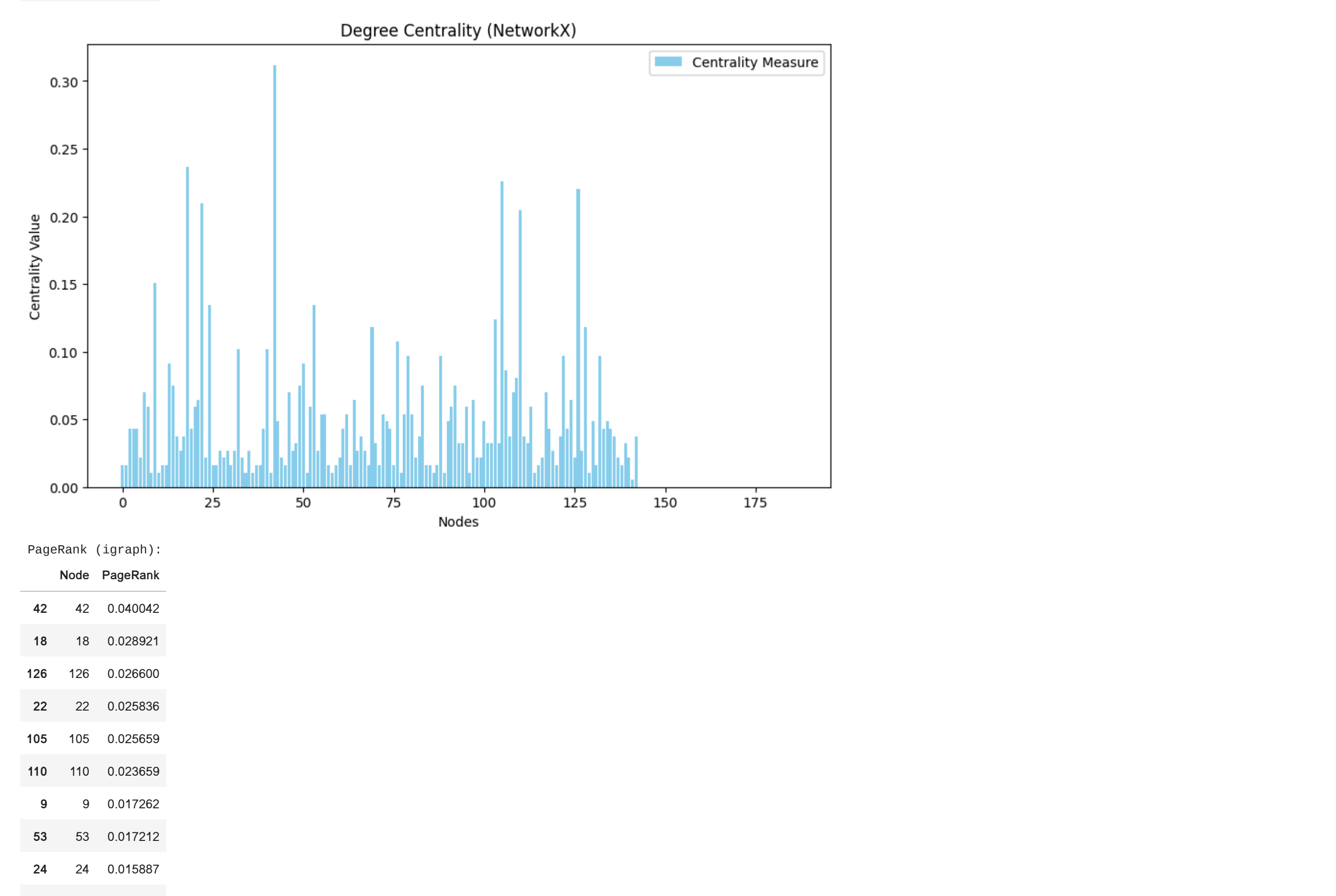
Closeness Centrality (NetworkX):

	Node	Closeness
42	42	0.463285
105	105	0.435376
18	18	0.433634
22	22	0.430193
126	126	0.426806
110	110	0.416956
9	9	0.392785
103	103	0.392785
24	24	0.392785
53	53	0.383069



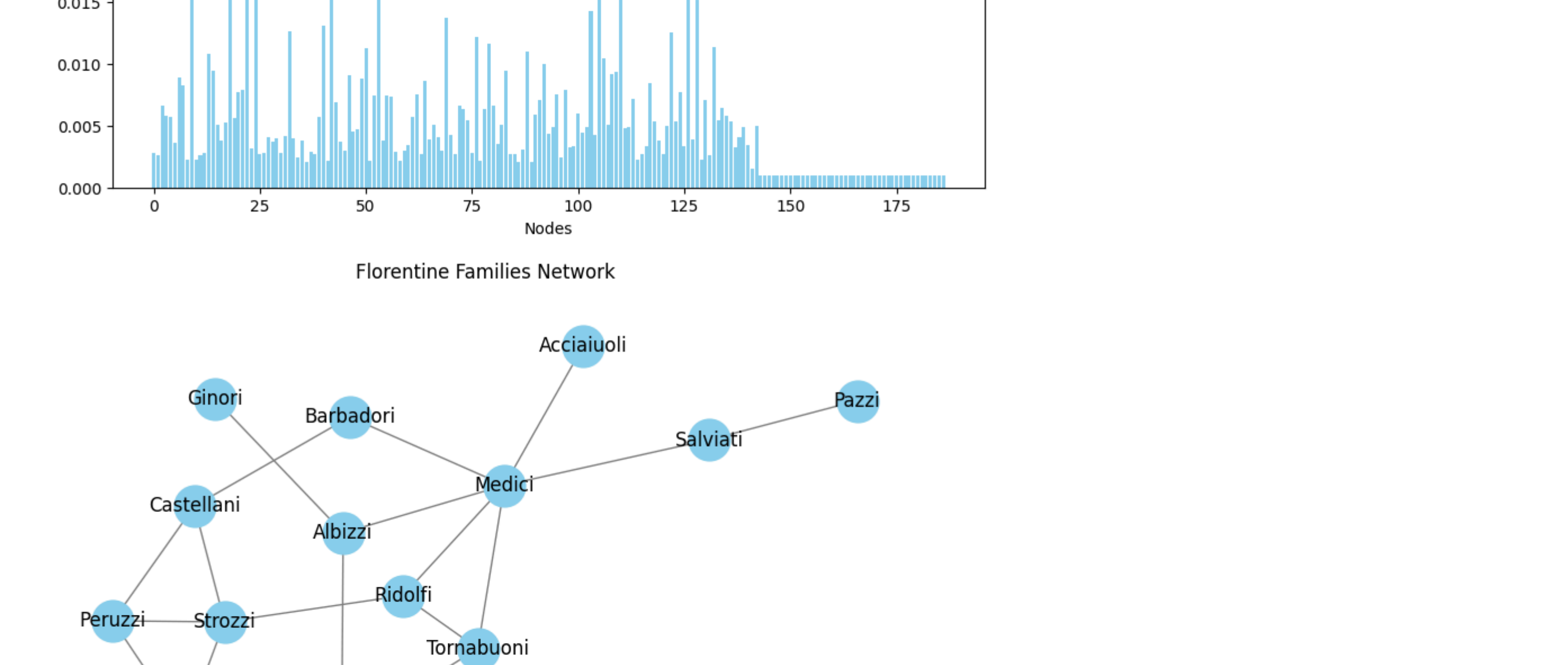
Degree Centrality (NetworkX):

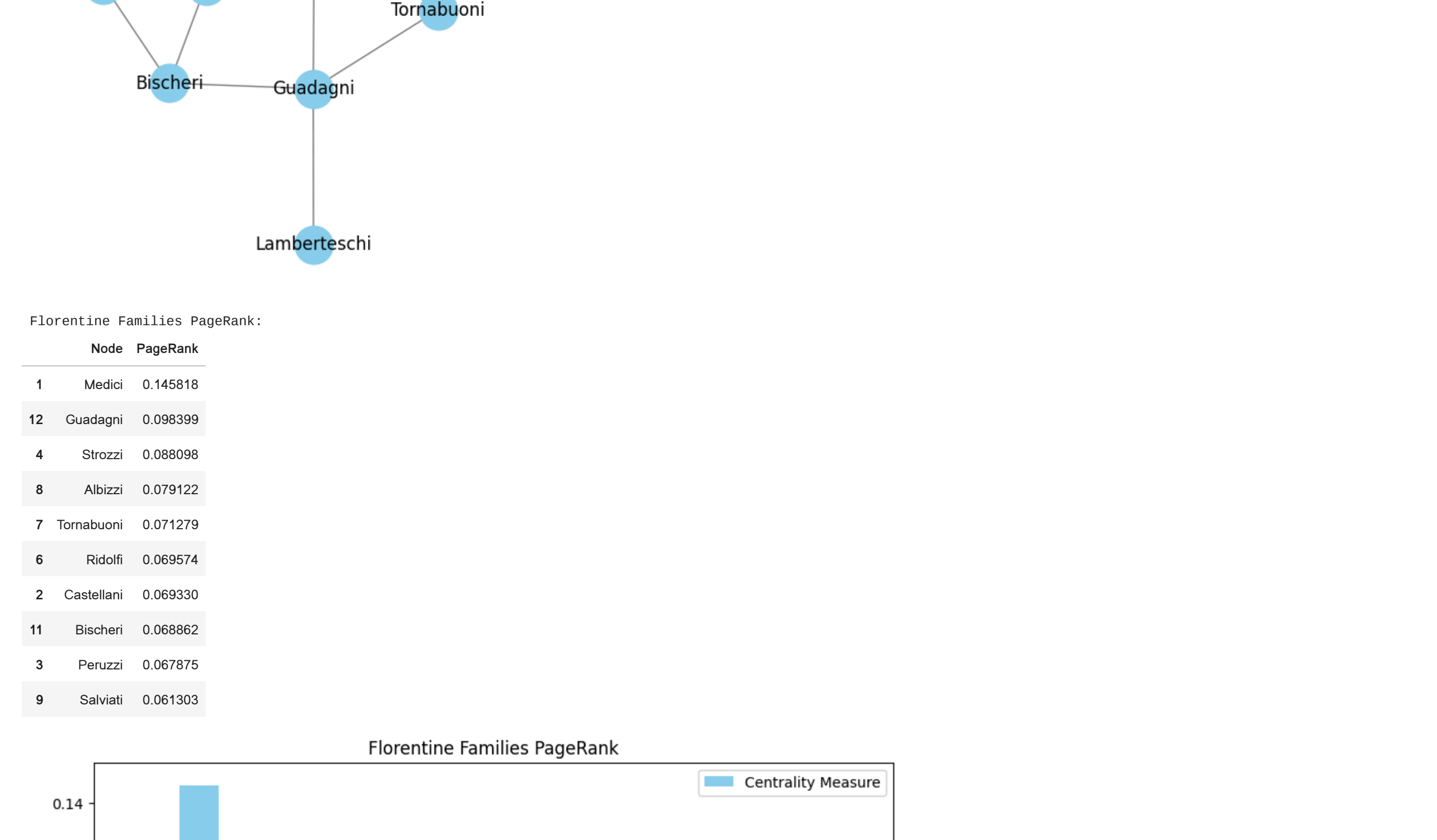
	Node	Degree
42	42	0.311828
18	18	0.236559
105	105	0.225806
126	126	0.220430
22	22	0.209677
110	110	0.204301
9	9	0.150538
24	24	0.134409
53	53	0.134409
103	103	0.123656



PageRank (igraph):

	Node	PageRank
42	42	0.040042
18	18	0.028921
126	126	0.026600
22	22	0.025836
105	105	0.025659
110	110	0.023659
9	9	0.017262
53	53	0.017212
24	24	0.015887
128	128	0.015146





Florentine Families PageRank:

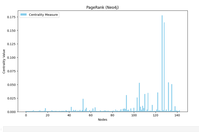
	Node	PageRank
1	Medici	0.145818
12	Guadagni	0.098399
4	Strozzi	0.088098
8	Albizzi	0.079122
7	Tornabuoni	0.071279
6	Ridolfi	0.069574
2	Castellani	0.069330
11	Bischeri	0.068862
3	Peruzzi	0.067875
9	Salviati	0.061303

PageRank (Neo4j):

	label	score
0	Yoren	4.369831
1	Albett	3.544865
2	Aegon I Targaryen	2.984199
3	Todder	2.074483
4	Tyrion Lannister	1.933146
5	Arthur Dayne	1.883858
6	Timett	1.301297
7	Balon Swann	1.209997
8	Lyn Corbray	1.187182
9	Aerys II Targaryen	1.181491

PageRank (Neo4j):

	Node	PageRank
130	126	0.177753
125	128	0.164763
137	132	0.054039
116	105	0.053055
129	135	0.050795
115	122	0.035313
99	113	0.034760
96	110	0.032574
100	93	0.030756
110	103	0.026108


PageRank (Node4j)
Centrality Measure
Centrality Score
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0
25
50
75
100
125
150
Nodes