一張含有 文字 的圖片

自動產生的描述

Marc Aurel Schnabel

import networkx as nx

import pandas as pd

import matplotlib.pyplot as plt

# 创建一个空的无向图

G = nx.Graph()

# 从Excel文件加载数据

excel\_file = "CADVis\_Author.xlsx" # 替换为您的Excel文件路径

df = pd.read\_excel(excel\_file)

# 根据相同的 work\_id 创建节点和边

for work\_id, group in df.groupby("work\_id"):

authors = group["Fullname"].tolist()

order = group["Order"].tolist()

# 添加节点（作者）

for author in authors:

G.add\_node(author) # 添加节点（作者）

# 添加带有作者顺序的边（合作关系）

for i in range(len(authors)):

for j in range(i + 1, len(authors)):

G.add\_edge(authors[i], authors[j], order=order[i]) # 使用order[i]来表示作者顺序

# 创建一个子图，只包括与Marc Aurel Schnabel相关的节点和边

subgraph = G.subgraph(["Marc Aurel Schnabel"] + list(G.neighbors("Marc Aurel Schnabel")))

# 计算节点的度中心性

degree\_centrality = nx.degree\_centrality(subgraph)

# 计算节点的大小，根据度中心性调整

node\_size = [300 \* degree\_centrality[node] for node in subgraph.nodes()]

# 可视化SNA图

plt.figure(figsize=(12, 12))

# 自定义布局

pos = nx.spring\_layout(subgraph, seed=42)

# 绘制边，根据作者顺序添加不同颜色的边（浅色系）

for edge in subgraph.edges(data=True):

order = edge[2]["order"]

if order == 1:

edge\_color = "lightcoral" # 浅红色

elif order == 2:

edge\_color = "lightblue" # 浅蓝色

elif order == 3:

edge\_color = "lightgreen" # 浅绿色

else:

edge\_color = "lightgray" # 浅灰色

nx.draw\_networkx\_edges(subgraph, pos, edgelist=[(edge[0], edge[1])], edge\_color=edge\_color, width=2)

# 绘制节点，节点大小基于度中心性，节点颜色为芥末色

nx.draw(subgraph, pos, node\_size=node\_size, node\_color='#A4C639', with\_labels=False, edgecolors='black', linewidths=0.5)

# 绘制节点标签（度中心性），但调整它们的位置以避免重叠，并设置文字颜色为深黑色

label\_pos = {k: (v[0], v[1] - 0.02) for k, v in pos.items()}

labels = {node: f"{node}\nDegree Centrality: {degree\_centrality[node]:.2f}" for node in subgraph.nodes()}

nx.draw\_networkx\_labels(subgraph, label\_pos, labels=labels, font\_size=8, font\_color='#000000')

plt.title("Collaboration Network (SNA) Graph with Author Order (Marc Aurel Schnabel)")

plt.axis('off') # 关闭坐标轴

plt.show()

轉成JSON檔

import networkx as nx

import pandas as pd

import matplotlib.pyplot as plt

import json

from networkx.readwrite import json\_graph

# Create an empty undirected graph

G = nx.Graph()

# Load data from the Excel file

excel\_file = "CADVis\_Author.xlsx" # Replace with your Excel file path

df = pd.read\_excel(excel\_file)

# Create nodes and edges based on the 'work\_id'

for work\_id, group in df.groupby("work\_id"):

authors = group["Fullname"].tolist()

order = group["Order"].tolist()

for author in authors:

G.add\_node(author)

for i in range(len(authors)):

for j in range(i + 1, len(authors)):

G.add\_edge(authors[i], authors[j], order=order[i])

# Create a subgraph with Marc Aurel Schnabel and related nodes

subgraph = G.subgraph(["Marc Aurel Schnabel"] + list(G.neighbors("Marc Aurel Schnabel")))

# Calculate node degree centrality

degree\_centrality = nx.degree\_centrality(subgraph)

# Calculate node sizes based on degree centrality

node\_size = [300 \* degree\_centrality[node] for node in subgraph.nodes()]

# Visualize the graph

plt.figure(figsize=(12, 12))

pos = nx.spring\_layout(subgraph, seed=42)

for edge in subgraph.edges(data=True):

order = edge[2]["order"]

if order == 1:

edge\_color = "lightcoral"

elif order == 2:

edge\_color = "lightblue"

elif order == 3:

edge\_color = "lightgreen"

else:

edge\_color = "lightgray"

nx.draw\_networkx\_edges(subgraph, pos, edgelist=[(edge[0], edge[1])], edge\_color=edge\_color, width=2)

nx.draw(subgraph, pos, node\_size=node\_size, node\_color='#A4C639', with\_labels=False, edgecolors='black', linewidths=0.5)

label\_pos = {k: (v[0], v[1] - 0.02) for k, v in pos.items()}

labels = {node: f"{node}\nDegree Centrality: {degree\_centrality[node]:.2f}" for node in subgraph.nodes()}

nx.draw\_networkx\_labels(subgraph, label\_pos, labels=labels, font\_size=8, font\_color='#000000')

plt.title("Collaboration Network (SNA) Graph with Author Order (Marc Aurel Schnabel)")

plt.axis('off')

plt.show()

# Export graph structure to JSON

graph\_data = json\_graph.node\_link\_data(subgraph)

with open('graph\_data.json', 'w') as json\_file:

json.dump(graph\_data, json\_file, indent=2)

# Export node attributes (degree centrality) to JSON

node\_attributes = {node: degree\_centrality[node] for node in subgraph.nodes()}

with open('node\_attributes.json', 'w') as json\_file:

json.dump(node\_attributes, json\_file, indent=2)