Grafo de Bolsistas de Iniciação Científica

Componentes:
Jucinara Melo
Pablo Arthur

Objetivo do Projeto

- Criar base de dados real a partir de bolsistas de IC;
- Transformar em **grafo bolsista ↔ orientador**;
- Analisar propriedades da rede;
- Visualizar em **Python** e **Gephi**.

Fontes de Dados

- Base CSV fornecida: `bolsistas-iniciacao-cientificia.csv`;
- Dados contêm: discentes, orientadores, projetos, unidades, status das bolsas;

Etapas do Trabalho

1. Limpeza e normalização de dados
2. Modelagem em banco SQLite
3. Exportação de CSVs
4. Construção do grafo
5. Análises
6. Visualização (Matplotlib e Gephi)

Limpeza e Modelagem

```python
# Padronizar nomes das colunas
df.columns = [c.strip().lower() for c in df.columns]

# Tratar strings em colunas de texto
text_cols = ["discente","orientador","titulo","linha_pesquisa","grupo_pesquisa",
             "unidade","categoria","tipo_de_bolsa","cota","status"]

for c in text_cols:
    if c in df.columns:
        df[c] = df[c].astype(str).str.strip().replace({"nan": None})

# Converter datas
for c in ["inicio","fim"]:
    if c in df.columns:
        df[c] = pd.to_datetime(df[c], dayfirst=True, errors="coerce")

# Converter ano
if "ano" in df.columns:
    df["ano"] = pd.to_numeric(df["ano"], errors="coerce").astype("Int64")

df.head()
```

Criar tabelas normalizadas

```
#Filtra as bolsas que estão em andamento
df_andamento = df[df['status'] == 'EM ANDAMENTO']

discentes = df_andamento[["id_discente","matricula","discente"]].drop_duplicates()
orientadores = df_andamento[["id_orientador","orientador"]].drop_duplicates()
unidades = df_andamento[["id_unidade","unidade"]].drop_duplicates()
grupos = df_andamento[["id_grupo_pesquisa","grupo_pesquisa"]].drop_duplicates()
projetos = df_andamento[["codigo_projeto","id_projeto_pesquisa","titulo","linha_pesquisa"]].drop_duplicates()

fato_cols = ["id_discente","id_orientador","codigo_projeto","id_unidade",
             "ano","tipo_de_bolsa","categoria","cota","inicio","fim","status"]

fato_cols = [c for c in fato_cols if c in df.columns]
bolsas = df_andamento[fato_cols].copy()

print("Discentes e Orientadores:", discentes.shape)
print("Orientadores:", orientadores.shape)
print("Projetos:", projetos.shape)
print("Bolsas:", bolsas.shape)
```

```
Discentes e Orientadores: (1628, 3)
Orientadores: (804, 2)
Projetos: (1636, 4)
Bolsas: (1659, 11)
```

Criar banco SQLite

```
import sqlite3

DB = "ic_bolsas.sqlite"
conn = sqlite3.connect(DB)

# Salvar tabelas no banco
discentes.to_sql("discentes", conn, if_exists="replace", index=False)
orientadores.to_sql("orientadores", conn, if_exists="replace", index=False)
unidades.to_sql("unidades", conn, if_exists="replace", index=False)
grupos.to_sql("grupos_pesquisa", conn, if_exists="replace", index=False)
projetos.to_sql("projetos", conn, if_exists="replace", index=False)
bolsas.to_sql("bolsas", conn, if_exists="replace", index=False)

# Criar índices para acelerar consultas
cur = conn.cursor()
cur.executescript("""
CREATE INDEX IF NOT EXISTS idx_bolsas_id_discente ON bolsas(id_discente);
CREATE INDEX IF NOT EXISTS idx_bolsas_id_orientador ON bolsas(id_orientador);
CREATE INDEX IF NOT EXISTS idx_bolsas_codigo_projeto ON bolsas(codigo_projeto);
CREATE INDEX IF NOT EXISTS idx_bolsas_id_unidade ON bolsas(id_unidade);
CREATE INDEX IF NOT EXISTS idx_bolsas_ano ON bolsas(ano);
""")
conn.commit()
conn.close()

print("Banco SQLite criado:", DB)
```

```
Banco SQLite criado: ic_bolsas.sqlite
```

Construção do Grafo

```
import networkx as nx
import matplotlib.pyplot as plt
```

```
# Lendo os CSVs exportados
discentes = pd.read_csv("exports/nodes_discentes.csv", encoding="utf-8-sig")
orientadores = pd.read_csv("exports/nodes_orientadores.csv", encoding="utf-8-sig")
projetos = pd.read_csv("exports/nodes_projetos.csv", encoding="utf-8-sig")
bolsas = pd.read_csv("exports/bolsas.csv", encoding="utf-8-sig")

# Mostrar as primeiras linhas de cada tabela
print("Discentes_orientadores")
display(discentes.head())

print("Orientadores")
display(orientadores.head())

print("Projetos")
display(projetos.head())

print("Bolsas")
display(bolsas.head())
```

	id_discente	matricula	discente
0	860404	20230026168	ADELSON DE OLIVEIRA DA CRUZ FILHO
1	877504	20240005395	ADILY MATHEUS DIAS DA SILVA
2	843056	20220039752	ADNA ELIZA TEIXEIRA SANTOS
3	823669	20210023875	ADRIANO IGHOR MOURA SOARES
4	830804	20210086356	AGATHA SOFIA BRITO DE ALMEIDA

Orientadores

	id_orientador	orientador
0	5752214	RICHARDSON AUGUSTO ROSENDO DA SILVA
1	5757936	MAGNO FRANCISCO DE JESUS SANTOS
2	5761206	MARCELO BRAZ MORAES DOS REIS
3	5763500	ISABEL CRISTINA DE OLIVEIRA MAGALHAES AMORIM
4	5753584	ROZELI MARIA PORTO

Projetos

	codigo_projeto	id_projeto_pesquisa	titulo	linha_pesquisa
0	PVD19942-2022	160310211	Alerta Vermelho: Risco de Sobrecarga de Estres...	Enfermagem na vigilância à saúde
1	PIC13027-2016	117639624	"Na linha de frente dos modernos escritores de...	Ensino de História e Patrimônio
2	PVE22287-2024	174896742	PLANO DE TRABALHO (2025-2026) REFERENTE À CONT...	Fundamentos do serviço social
3	PIG22307-2024	174920352	Análise de práticas e de projetos pedagógicos ...	Hábitos de Transporte e Desenvolvimento Susten...
4	PVC17966-2020	141841331	Dez anos da epidemia de Zika Vírus no Brasil	Antropologia do corpo, gênero e sexualidade

Bolsas

	id_discente	id_orientador	codigo_projeto	id_unidade	ano	tipo_de_bolsa	categoria	cota	inicio	fim	status
0	860404	5752214	PVD19942-2022	198	2022	PIBIC CNPq	Iniciação Científica (IC)	2025-2026 (PIBIC)	2025-02-09	2026-08-31 00:00:00.000	EM ANDAMENTO
1	877504	5757936	PIC13027-2016	141	2016	PIBIC UFRN	Iniciação Científica (IC)	2025-2026 (PIBIC)	2025-01-09	2026-08-31 00:00:00.000	EM ANDAMENTO
2	843056	5761206	PVE22287-2024	162	2024	PIBIC UFRN	Iniciação Científica (IC)	2025-2026 (PIBIC)	2025-05-09	2026-08-31 00:00:00.000	EM ANDAMENTO
3	823669	5763500	PIG22307-2024	52	2024	PIBIC CNPq	Iniciação Científica (IC)	2025-2026 (PIBIC)	2025-03-09	2026-08-31 00:00:00.000	EM ANDAMENTO
4	830804	5753584	PVC17966-2020	144	2020	PIBIC UFRN	Iniciação Científica (IC)	2025-2026 (PIBIC)	2025-01-09	2026-08-31 00:00:00.000	EM ANDAMENTO

Criar Grafo

```python
# Criar o grafo vazio
G = nx.Graph()

for _, row in df_andamento.iterrows():
    orientador = row['orientador']
    bolsista = row['discente']
    G.add_edge(orientador, bolsista)

print(f"Grafo criado com {G.number_of_nodes()} nós e {G.number_of_edges()} arestas.")
print("Nós no grafo:", G.nodes())
print("-" * 30)
```

Seleção dos N professores com mais bolsistas

```python
# Calcule o grau de cada nó
degrees = dict(G.degree())
df_degrees = pd.DataFrame(degrees.items(), columns=['Nó', 'Grau']).sort_values('Grau', ascending=False)

# Identifique quem são os orientadores
orientadores = df['orientador'].unique()
df_degrees['Tipo'] = df_degrees['Nó'].apply(lambda x: 'Orientador' if x in orientadores else 'Bolsista')

# Defina quantos orientadores com mais bolsistas você quer incluir no subgrafo
N = 15
top_orientadores = df_degrees[df_degrees['Tipo'] == 'Orientador'].head(N)['Nó'].tolist()

print(f"Os {N} orientadores mais conectados são:")
print(top_orientadores)
print("-" * 30)
```

Criação do subgrafo

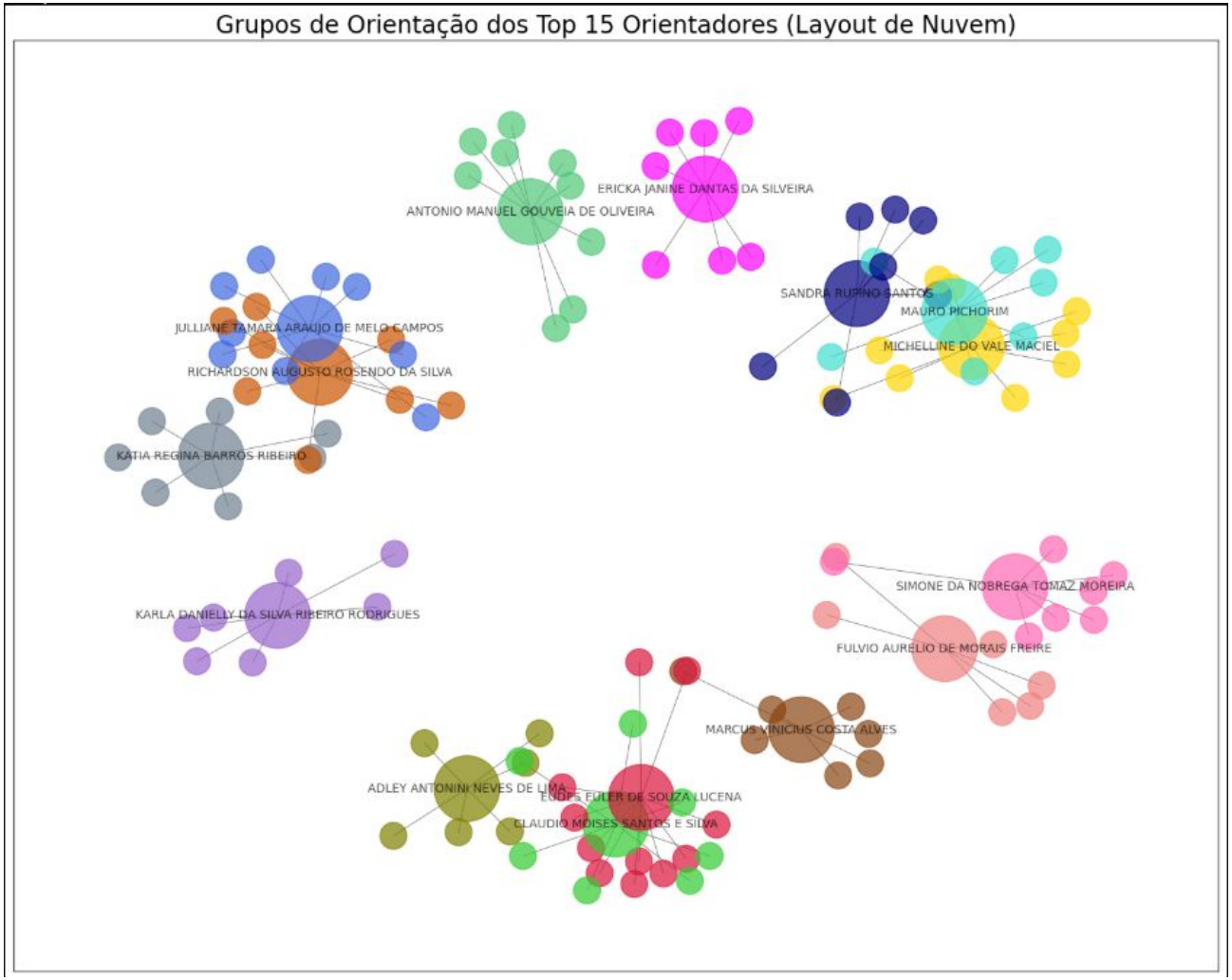
```
# Cria um subgrafo
nos_para_subgrafo = set(top_orientadores)
for orientador in top_orientadores:
    alunos_deste_orientador = G.neighbors(orientador) # Pega os alunos do orientador
    nos_para_subgrafo.update(alunos_deste_orientador) # Adiciona no conjunto do subgrafo
subgrafo_top_N = G.subgraph(nos_para_subgrafo) # Cria um subgrafo de G com o novo conjunto

#Atribui a cada nó professor uma cor
mapa_de_cores = {orientador: cores_hex[i % len(cores_hex)]
                 for i, orientador in enumerate(top_orientadores)}
```

Atributos

```python
# Itere sobre os nós para definir TODOS os atributos de uma vez
for node in subgrafo_top_N.nodes():
    if node in top_orientadores:
        subgrafo_top_N.nodes[node]['tipo'] = 'Orientador'
        subgrafo_top_N.nodes[node]['size'] = 3000
        subgrafo_top_N.nodes[node]['color'] = mapa_de_cores[node]

    else:
        professor = list(subgrafo_top_N.neighbors(node))[0]
        subgrafo_top_N.nodes[node]['tipo'] = 'Bolsista'
        subgrafo_top_N.nodes[node]['size'] = 500
        subgrafo_top_N.nodes[node]['color'] = mapa_de_cores[professor]
```

Plot do grafo

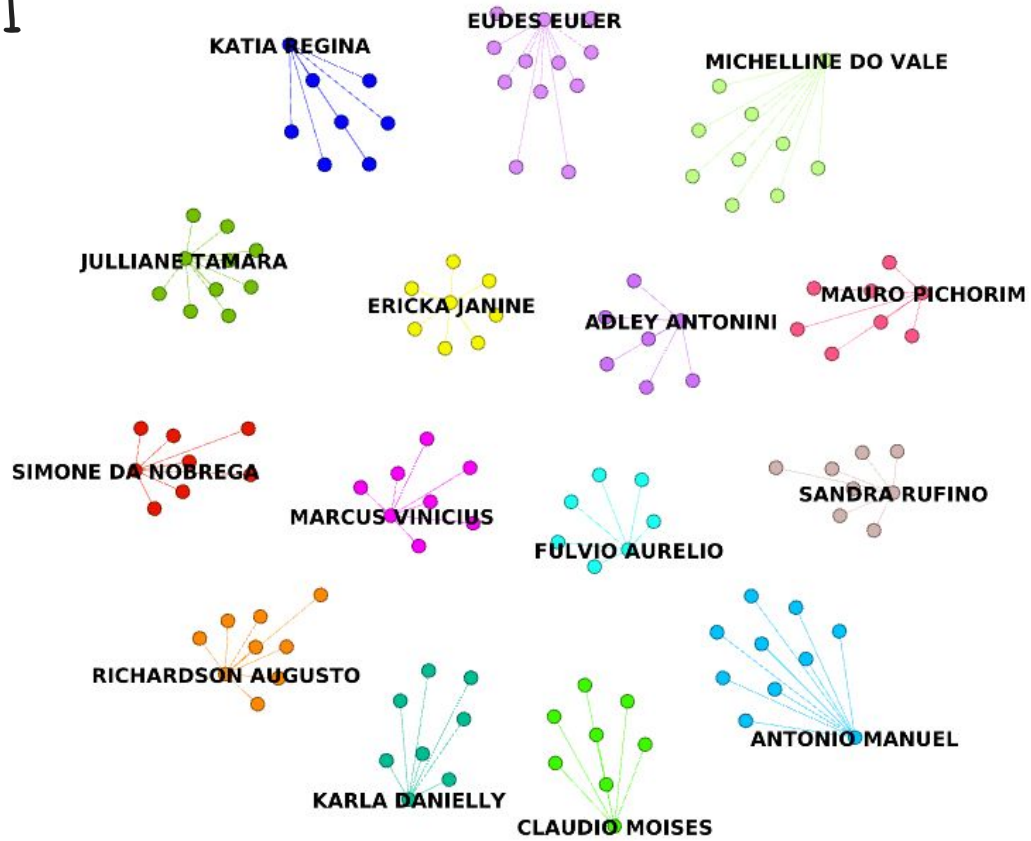
```python
# Atributos que acabamos de salvar no grafo para a visualização
node_colors = [subgrafo_top_N.nodes[n]['color'] for n in subgrafo_top_N.nodes()]
node_sizes = [subgrafo_top_N.nodes[n]['size'] for n in subgrafo_top_N.nodes()]
nomes_orientadores = {n: n for n in subgrafo_top_N.nodes() if subgrafo_top_N.nodes[n]['tipo'] == 'Orientador'}
posicoes = nx.spring_layout(subgrafo_top_N, iterations=100, k=0.4)

# Desenhando o grafo
plt.figure(figsize=(18, 14))

nx.draw_networkx(
    subgrafo_top_N,
    pos=posicoes,
    labels=nomes_orientadores,
    with_labels=True,
    node_size=node_sizes,
    node_color=node_colors,
    width=0.5,
    alpha=0.7,
    font_size=10
)

plt.title(f'Grupos de Orientação dos Top {N} Orientadores (Layout de Nuvem)', size=20)
plt.show()
```


Grupos de Orientação dos Top 15 Orientadores (Layout de Nuvem)
ANTONIO MANUEL GOUVEIA DE OLIVEIRA
ERICKA JANINE DANTAS DA SILVEIRA
SANDRA RUFINO SANTOS
MAURO PICHORIM
MICHELLINE DO VALE MACIEL
JULLIANE TAMARA ARAUJO DE MELO CAMPOS
RICHARDSON AUGUSTO ROSENDO DA SILVA
KATIA REGINA BARROS RIBEIRO
KARLA DANIELLY DA SILVA RIBEIRO RODRIGUES
SIMONE DA NOBREGA TOMAZ MOREIRA
FULVIO AURELIO DE MORAIS FREIRE
MARCUS VINICIUS COSTA ALVES
ADLEY ANTONINI NEVES DE LIMA
ENDYS FULLER DE SOUZA LUCENA
CLAUDIO MOISES SANTOS E SILVA

Salvando o arquivo gexf

```
# --- SALVAR O ARQUIVO GEXF ---
print("Salvando grafo enriquecido para o Gephi...")
nx.write_gexf(subgrafo_top_N, '1_gephi.gexf')
print("Arquivo salvo!")
```

GRAFO NO GEPHI



Layout force atlas

CONCLUSÃO

- Projeto mostrou aplicabilidade de **grafos em dados reais**;
- Python → pré-processamento, modelagem e análise;
- Gephi → exploração interativa e estética da rede;
- Abriu caminho para investigações futuras (ex.: comunidades acadêmicas, evolução temporal).

OBRIGADO(A)!