UNIVERSIDADE FEDERAL DO MARANHÃO

CIÊNCIA DA COMPUTAÇÃO – DEIN0083

Willy Kauã Diniz Teixeira

ATIVIDADE PRÁTICA - UNIDADE III

São Luís – MA

04 de Janeiro, 2026

INTRODUÇÃO

Este trabalho tem como objetivo desenvolver um sistema de recomendação de amizades para uma rede social fictícia chamada *SocialNet*, utilizando grafos como estrutura de dados principal. Os usuários são representados como vértices e as conexões de amizade como arestas, possibilitando a aplicação de algoritmos de análise de grafos para recomendação de novas conexões, identificação de comunidades e detecção de usuários influentes. A implementação foi realizada em Java, explorando conceitos como busca em largura, busca em profundidade e componentes conexos, com o intuito de consolidar o aprendizado prático sobre grafos e sua aplicação em redes sociais.

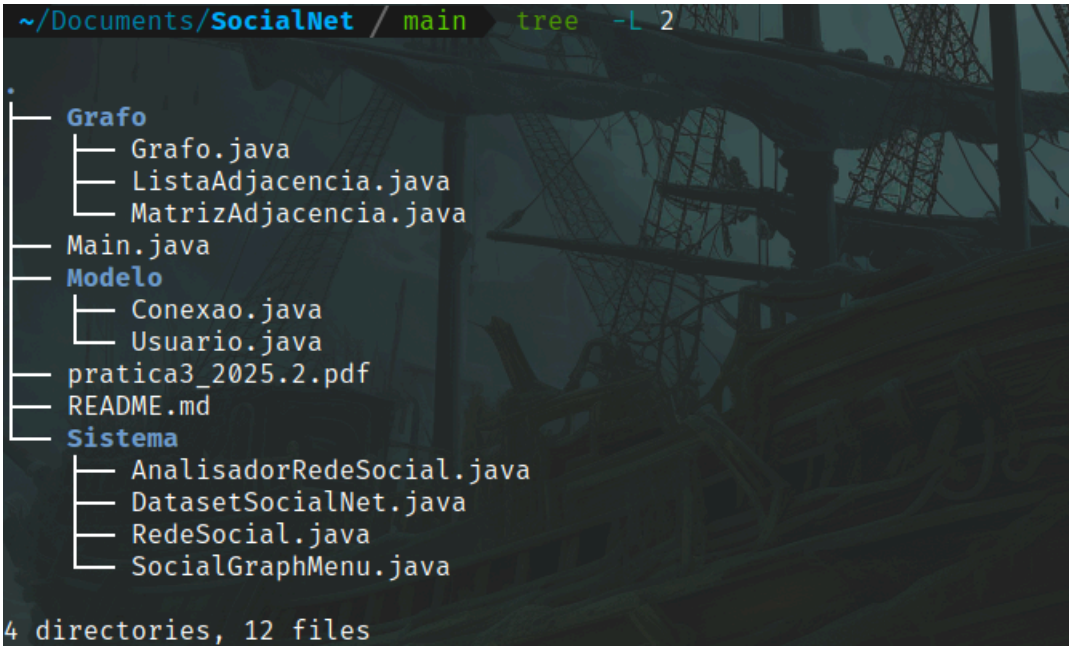
Link para apresentação:

https://drive.google.com/drive/folders/1a2DBWrYeGbhp-gPGBC3t2qLYxQlobGy7?usp=sharing

MODELAGEM DA REDE SOCIAL

A rede social foi modelada utilizando grafos, nos quais cada usuário é representado por um vértice e cada amizade por uma aresta ponderada, sendo o peso responsável por indicar a força da conexão entre dois usuários. Para permitir flexibilidade e comparação, foram implementadas duas representações de grafo: lista de adjacência e matriz de adjacência, ambas seguindo a interface genérica Grafo<T>. A gestão da rede é realizada pela classe RedeSocial, responsável pelo cadastro de usuários, adição e remoção de amizades e visualização das conexões. As análises da rede, como recomendação de amizades, identificação de comunidades, cálculo de influência e caminhos entre usuários, foram centralizadas na classe AnalisadorRedeSocial, enquanto a interação com o sistema é feita por meio de um menu textual implementado na classe SocialGraphMenu.

A estrutura do projeto está dividida da seguinte forma:



TESTES E RESULTADOS

Os testes do sistema foram realizados a partir de um dataset estático de usuários e conexões, previamente definido em uma classe específica do projeto. Esse conjunto de dados foi carregado automaticamente na inicialização da aplicação, permitindo a execução controlada e reprodutível das funcionalidades implementadas. Dessa forma, foi possível validar o comportamento dos algoritmos da rede social em diferentes cenários, como recomendação de amizades, identificação de comunidades e análise de influência entre usuários.

O programa fornece uma interface interativa no terminal com as seguintes opções:

```
=== SOCIALNET ===
1 - Cadastrar usuário
2 - Adicionar amizade
3 - Recomendar amizades
4 - Visualizar rede
5 - Encontrar caminho entre usuários
6 - Analises da rede
0 - Sair
Escolha uma opção:
```

Visualização de rede

Cada usuário deve ser exibido seguido de seus amigos diretos, confirmando que as arestas do grafo foram corretamente estabelecidas no dataset inicial. Selecionando a opção 4, temos:

```
=== Visualizar Rede ===
Felipe -> Ana
Gabriela -> Ana
Henrique ->
Isabela ->
Ursula -> Victor Rafael
Tiago -> Sofia
Wesley -> Victor Xavier
Victor -> Ursula Wesley
Yasmin -> Xavier Zeca
Xavier -> Wesley Yasmin
Alice -> Zeca Breno
```

OBS: A captura de tal não cobriu a exibição completa.

Recomendação de amizade

O sistema deve sugerir usuários que não sejam amigos diretos, priorizando aqueles com maior número de amigos em comum. Selecionando a opção 3, e escolhendo o usuário "u1" (por ID, cujo nome é Ana), temos o resultado:

```
=== Recomendar Amizades ===
ID do usuário: u1
Sugestões de amizade:
- Karla (ID: u11)
- Eduarda (ID: u5)
```

Busca de caminho entre usuários

O sistema deve exibir um caminho válido quando existir conexão entre os usuários, ou informar corretamente quando não houver caminho. Já que analisamos que Eduarda é uma recomendação de amizade, procuramos o caminho até ela:

```
ID do usuário origem: u1
ID do usuário destino: u5
Caminho de conexão:
Ana -> Bruno -> Eduarda
```

O que faz sentido, quando vemos que Bruno é um amigo em comum entre Ana e Eduarda:

```
Bruno -> Ana Carla Eduarda
```

Adicionando amizade

Aproveitando a recomendação, vamos adicionar Eduarda como uma amizade de Bruna, e visualizar a rede novamente:

```
=== Adicionar Amizade ===
ID do usuário origem: u1
ID do usuário destino: u5
Força da conexão (0.0 a 1.0): 1
Amizade adicionada com sucesso!
```

Resultado:

```
Ana -> Felipe Gabriela Bruno Carla Daniel Eduarda
Quezia -> Paulo Rafael
Bruno -> Ana Carla Eduarda
Carla -> Ana Bruno
Sofia -> Tiago Rafael
Daniel -> Karla Ana
Eduarda -> Ana Bruno
```

Identificação de comunidades

O sistema possui uma sub-interface para análises da rede, para visualizar as opções, selecionamos o caso 6:

```
=== ANÁLISES DA REDE ===
1 - Identificar comunidades
2 - Mostrar influenciadores
3 - Calcular distância social
4 - Verificar ciclos
5 - Encontrar pontes
6 - Gerar rede essencial
0 - Voltar
Escolha uma opção:
```

O sistema deve listar diferentes comunidades, agrupando usuários conectados direta ou indiretamente, e separar usuários isolados em comunidades individuais.

A primeira comunidade é uma lista bem extensa, contendo 27/30 usuários, pois todos eles se conectam de alguma forma. O que sobraram foram apenas usuários isolados, que não possuem amizades com ninguém.

Mostrar influenciadores

Os usuários retornados devem possuir maior número de conexões em relação aos demais, e o grau deve ser exibido corretamente.

```
Quantidade de influenciadores (top N): 3
Top 3 influenciadores:
Ana (grau: 6)
Rafael (grau: 4)
Lucas (grau: 3)
```

Calcular distância social

O sistema deve retornar um mapa contendo todos os usuários alcançáveis e suas respectivas distâncias, considerando a força das conexões. Usamos a Ana como exemplo, segue alguns dos resultados:

```
ID origem: u1
Ana -> Felipe: 0.19999999999999996
Ana -> Gabriela: 0.09999999999999998
Ana -> Henrique: Infinity
Ana -> Isabela: Infinity
Ana -> Ursula: 2.4000000000000004
Ana -> Tiago: 2.5000000000000004
Ana -> Wesley: 2.9000000000000004
Ana -> Victor: 2.6000000000000005
Ana -> Yasmin: 3.8000000000000003
```

Verificar ciclos

O sistema deve indicar a presença de ciclos, considerando que a rede social contém relações bidirecionais. No nosso exemplo, o sistema identificou que a rede possui ciclos.

Encontrar pontes

O sistema deve listar as conexões identificadas como pontes, incluindo os usuários envolvidos.

```
Ana - Gabriela
Clara - Diego
Breno - Clara
Alice - Breno
Zeca - Alice
Yasmin - Zeca
Xavier - Yasmin
Wesley - Xavier
Victor - Wesley
Ursula - Victor
Rafael - Ursula
Sofia - Tiago
Rafael - Sofia
Lucas - Paulo
Daniel - Karla
Ana - Daniel
Felipe - Ana
```

Gerar rede essencial

O sistema deve gerar um grafo que conecte todos os usuários com o menor custo possível, sem ciclos. Para o nosso exemplo, o sistema criou um grafo com 26 conexões.

CONCLUSÃO

O projeto permitiu aplicar, de forma prática, conceitos fundamentais de grafos na modelagem e análise de uma rede social. A utilização de algoritmos clássicos possibilitou a implementação de funcionalidades como recomendação de amizades, identificação de comunidades, análise de influência e verificação de conectividade entre usuários. Os testes realizados com um dataset estático demonstraram que o sistema se comporta de maneira consistente, refletindo características esperadas de redes sociais reais, como a formação de grandes componentes conexas e a presença de usuários isolados. Dessa forma, o trabalho atingiu seus objetivos ao integrar teoria e prática em uma aplicação funcional e bem estruturada.