UNIVERSIDADE FEDERAL DE LAVRAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Aluno: Michel Alexandrino de Souza **Matrícula: 202111084**

GCC218 – Algoritmos em Grafos Prof. Vinicius Vitor dos Santos Dias

Relatório Técnico

1. Definição e apresentação da base de dados

Ao desenvolver a solução para o problema de recomendação de filmes em Python, encontrei desafios que impactaram o processo. A necessidade de aprender e aprimorar minha compreensão da estrutura da linguagem Python e me familiarizar com o Windows Subsystem for Linux (WSL) consumiu tempo significativo. Além disso, a gestão de uma base de dados extensa em um computador com limitações de hardware resultou em demoras na compilação do código. Para contornar esse problema, limitei o acesso aos dados, mas essa abordagem levantou questões sobre a representatividade das recomendações, dada a redução no conjunto de dados. A escolha da filtragem colaborativa mostrou-se eficaz, mas a análise crítica destaca áreas de melhoria, especialmente em relação à gestão de grandes conjuntos de dados e à representatividade das recomendações geradas. Embora o algoritmo tenha oferecido sugestões personalizadas, a limitação da base de dados pode ter impactado a diversidade e precisão das recomendações, ressaltando a importância de equilibrar a eficiência computacional com a representatividade dos resultados. Quando se trata dos resultados obtidos, vale ressaltar que, para interpretar os dados, seguimos uma abordagem prática. Selecionamos aleatoriamente um ID da base de dados de usuários e, usando o algoritmo de filtragem colaborativa, geramos uma lista de filmes recomendados para esse usuário específico. Essa escolha deliberada de um ID aleatório na base de dados visa fornecer uma visão mais tangível da eficácia do sistema, proporcionando recomendações personalizadas para diferentes perfis de usuários. Essa abordagem contribui para uma análise mais abrangente das capacidades do algoritmo de recomendação em diversos cenários.

2. Limitação do escopo e definição do problema

Ao direcionar a atenção para a recomendação de filmes, o problema a ser abordado envolve a seleção de um usuário específico identificado pelo seu ID na base de dados. A entrada para essa análise compreende o ID de um usuário escolhido, os IDs dos filmes que ele avaliou e as arestas que conectam esses vértices, representando as respectivas notas atribuídas pelo usuário aos filmes. A saída esperada consiste em uma mensagem indicando que estão sendo recomendados filmes para outro usuário previamente escolhido na base de dados. Essa mensagem é acompanhada por uma lista contendo os IDs dos filmes recomendados com base nas conexões do grafo. No caso de não ser possível realizar recomendações para o usuário em questão, uma mensagem de erro será exibida. Além disso, se não for possível localizar o usuário selecionado na base de dados, uma mensagem de erro adicional será apresentada. Essa abordagem visa tornar a análise do grafo uma solução prática e interativa, fornecendo recomendações personalizadas com

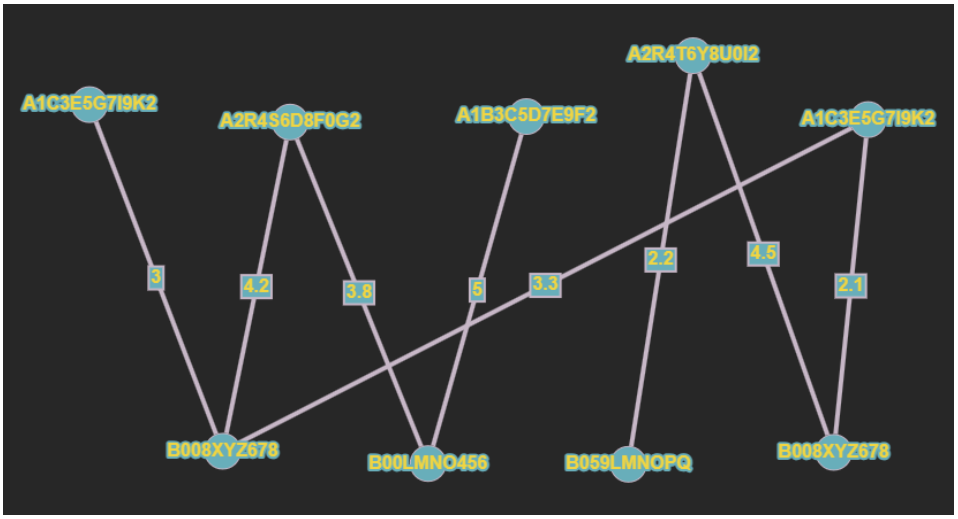
base nas interações presentes na base de dados, e assegurando respostas claras e específicas para diferentes cenários.

3. Solução proposta e ferramentas utilizada

A solução que desenvolvi para o problema de recomendação de filmes é implementada em Python e construída utilizando a IDE Visual Studio Code, em conjunto com o subsistema Windows Subsystem for Linux (WSL). No código, optei por um algoritmo simples de filtragem colaborativa de ordem O(n * m), estruturando-o de maneira modular com classes específicas. A classe Dados cuida da leitura e pré-processamento dos dados do arquivo "movies.txt", organizando as informações de maneira adequada. Em seguida, a classe Grafo utiliza a biblioteca NetworkX para criar um grafo não direcionado, representando usuários, filmes e suas interações. A implementação do algoritmo de filtragem colaborativa é feita na classe Recomendador, que pondera as pontuações de usuários similares para gerar recomendações personalizadas. A interação com o usuário é facilitada pela classe Main, que oferece opções como visualizar dados do grafo, explorar informações de usuários aleatórios e receber recomendações de filmes. Embora a solução atual se concentre na filtragem colaborativa simples, ela pode ser expandida para incluir técnicas adicionais, como a filtragem baseada em conteúdo. Além disso, a escalabilidade do código pode ser otimizada para lidar com conjuntos de dados mais extensos. Para rodar este código, basta utilizar o comando "make" no terminal. Em resumo, a metodologia adotada, juntamente com a escolha de Python, VS Code e WSL, visa fornecer uma solução eficiente, modular e passível de expansão para o desafio de recomendação de filmes.

4. Resultados e limitações

Ao desenvolver a solução para o problema de recomendação de filmes em Python, encontrei desafios que impactaram o processo. A necessidade de aprender e aprimorar minha compreensão da estrutura da linguagem Python e me familiarizar com o Windows Subsystem for Linux (WSL) consumiu tempo significativo. Além disso, a gestão de uma base de dados extensa em um computador com limitações de hardware resultou em demoras na compilação do código. Para contornar esse problema, limitei o acesso aos dados, mas essa abordagem levantou questões sobre a representatividade das recomendações, dada a redução no conjunto de dados. A escolha da filtragem colaborativa mostrou-se eficaz, mas a análise crítica destaca áreas de melhoria, especialmente em relação à gestão de grandes conjuntos de dados e à representatividade das recomendações geradas. Embora o algoritmo tenha oferecido sugestões personalizadas, a limitação da base de dados pode ter impactado a diversidade e precisão das recomendações, ressaltando a importância de equilibrar a eficiência computacional com a representatividade dos resultados. Quando se trata dos resultados obtidos, vale ressaltar que, para interpretar os dados, seguimos uma abordagem prática. Selecionamos aleatoriamente um ID da base de dados de usuários e, usando o algoritmo de filtragem colaborativa, geramos uma lista de filmes recomendados para esse usuário específico. Essa escolha deliberada de um ID aleatório na base de dados visa fornecer uma visão mais tangível da eficácia do sistema, proporcionando recomendações personalizadas para diferentes perfis de usuários. Essa abordagem contribui para uma análise mais abrangente das capacidades do algoritmo de recomendação em diversos cenários.



Exemplo de grafo bipartido com K5,4 — vértices de cima são usuários e de baixo filmes.

```
soumichel@LAPTOP-2PH0SLPK:~/mini-projeto/src$ python3 main.py
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 1
Quantidade de nós: 9153
Quantidade de arestas: 9918
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 2
ID do usuário aleatório: A1MIFONBFK8OWE
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 3
Filmes recomendados para o usuário A4CZG36S2QV2T:
- A328S9RN3U5M68
- A18758S1PUYIDT
- A1EMDSTJDUE6B0
- ANCOMAI0I7LVG
- AQZH7YTWQPOBE
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 0
Encerrando o programa.
```

```
soumichel@LAPTOP-2PH0SLPK:~/mini-projeto/src$ python3 main.py
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 1
Quantidade de nós: 4784
Quantidade de arestas: 4957
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 2
ID do usuário aleatório: A1XUG3EJX89KNG
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 3
Filmes recomendados para o usuário A3LRVGJ73EMPQV:
- A328S9RN3U5M68
- A18758S1PUYIDT
- A1EMDSTJDUE6B0
- ANCOMAI0I7LVG
- AQZH7YTWQPOBE
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 0
Encerrando o programa.
```

```
soumichel@LAPTOP-2PH0SLPK:~/mini-projeto/src$ python3 main.py
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 1
Quantidade de nós: 6583
Quantidade de arestas: 6938
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 2
ID do usuário aleatório: A2HKY17EEOYZEB
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 3
Filmes recomendados para o usuário AEQFYOI6YJ83Z:
- A2582KMXLK2P06
- A2RW7HXNEJ0QTQ
- AUM3YMZ0YRJE0
- A9RNMO9MUSMTJ
- A35ZK3M8L9JUPX
1. Visualizar dados do grafo
2. Visualizar dados de um usuário aleatório na base de dados
3. Recomendar filmes para um usuário aleatório
0. Sair
Escolha uma opção (0-3): 0
Encerrando o programa.
```