UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE COMPUTAÇÃO

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GULLIT DAMIÃO TEIXEIRA DE CAMPOS

Busca em Largura (BFS)

Uberlândia, Brasil

2024.

GULLIT DAMIÃO TEIXEIRA DE CAMPOS

Busca em Largura (BFS)

Primeiro Projeto de conclusão da disciplina Inteligência Artificial apresentado ao Departamento da Faculdade de Computação da Universidade Federal de Uberlândia, de Uberlândia, Câmpus Santa Mônica, como parte dos requisitos para obtenção da pontuação referente a trabalhos e projeto da disciplina do curso bacharelado em Ciências de Computação.

Orientador: Prof. Dr. Jefferson Rodrigo de Souza

Uberlândia, Brasil

2024.

Problemática:

O código tem como objetivo simular a relação entre filmes do Universo Cinematográfico da Marvel (MCU) como um grafo, utilizando a técnica de **Busca em Largura (BFS)** com backtracking para encontrar o menor caminho entre dois filmes. Além disso, ele visualiza essas conexões de maneira gráfica para facilitar a compreensão da estrutura.

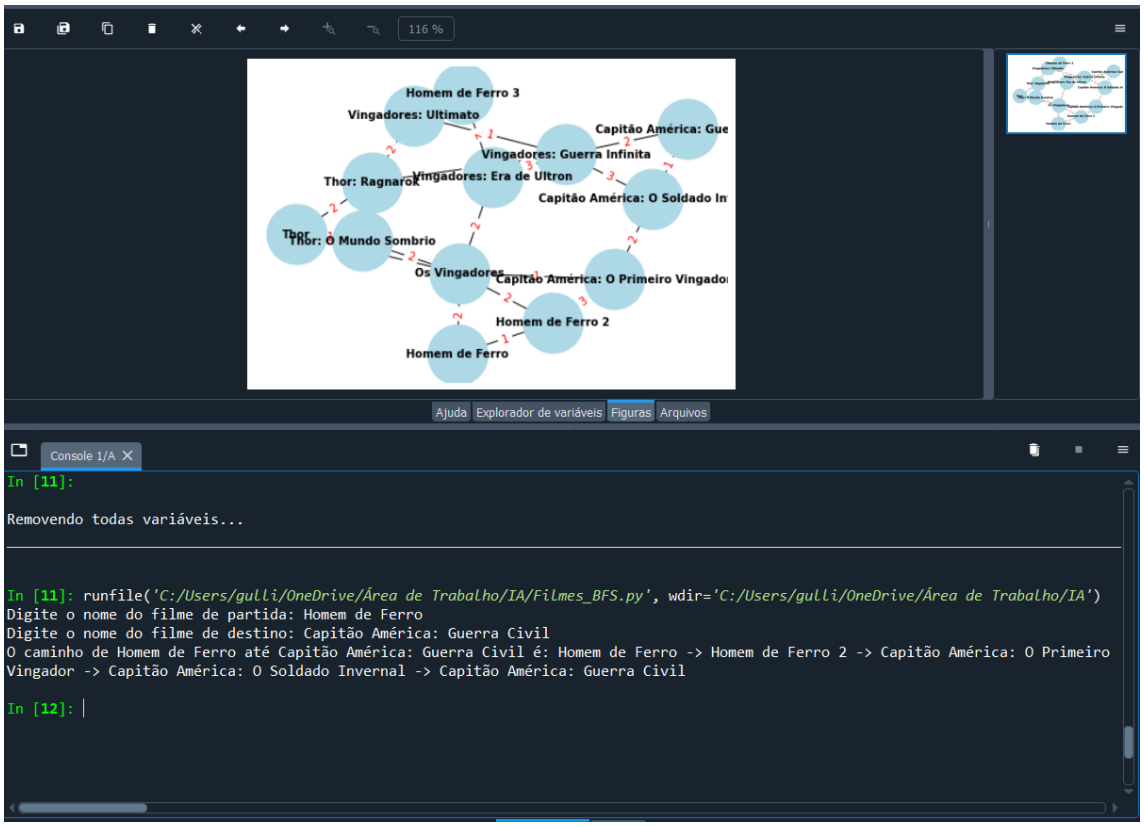
Objetivo: O objetivo do código é construir e explorar um grafo de filmes do MCU, no qual cada filme é um nó e as conexões entre eles (com base em relações ou aparições conjuntas) são as arestas. Mais especificamente, o código permite ao usuário:

Encontrar o menor caminho entre dois filmes específicos usando o algoritmo de **Busca em Largura (BFS)**, que é eficiente para esse tipo de tarefa em grafos não ponderados ou com pesos constantes.

Visualizar o grafo dos filmes e suas conexões, com o peso das arestas indicando a "relevância" ou "proximidade" da relação entre os filmes.

Realizar uma busca interativa, onde o usuário escolhe um filme inicial e um filme de destino, e o código retorna o menor caminho entre esses dois filmes (se existir).

Método: O código utiliza várias técnicas e conceitos importantes de algoritmos de grafos e visualização de dados.



1. Construção do Grafo

O código representa os filmes como um grafo, onde:

- Cada filme é um nó (representado como uma chave no dicionário lista_de_filmes).
- As conexões entre os filmes (arestas) são armazenadas como listas de tuplas no formato (filme_vizinho, peso).

Por exemplo:

'Os Vingadores': [('Homem de Ferro', 2), ('Thor', 1), ('Capitão América: O Primeiro Vingador', 1), ('Vingadores: Era de Ultron', 2)],

'Os Vingadores' está conectado a outros filmes como 'Homem de Ferro', 'Thor', ..., com um peso associado a cada conexão. Esse peso pode representar a relevância ou a proximidade temporal dos filmes na linha do tempo do MCU.

2. Busca em Largura (BFS)

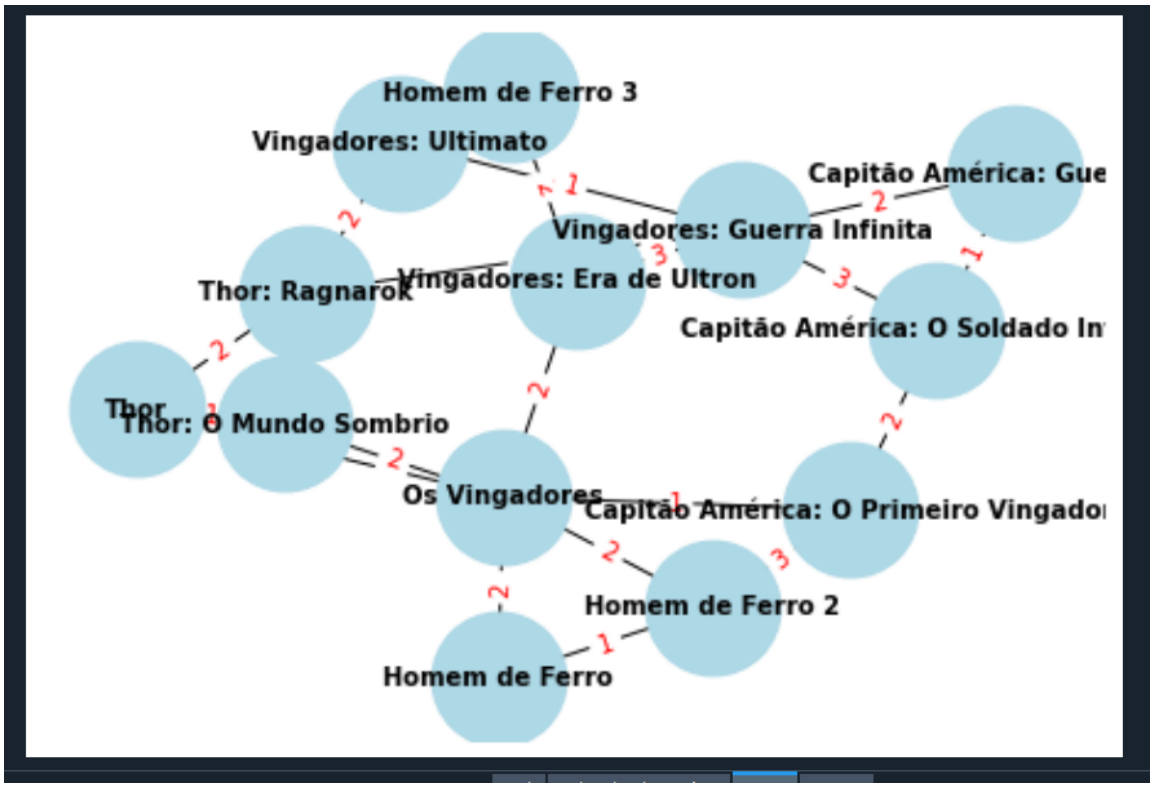
O método principal de busca é o **BFS** (Busca em Largura), que é um algoritmo usado para explorar grafos e encontrar o menor caminho (em termos de número de arestas) entre dois nós. No código, a função bfs_backtracking executa essa busca com os seguintes passos:

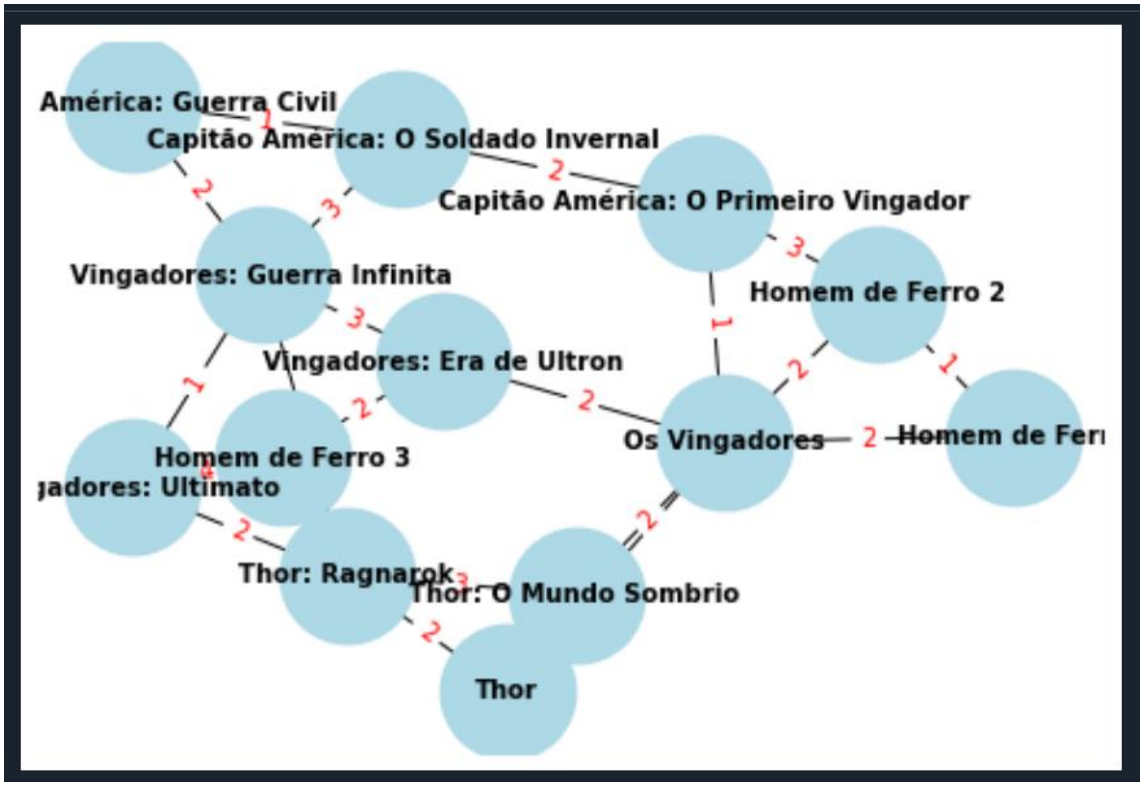
Fila de Exploração: Uma fila é usada para manter os nós (filmes) que precisam ser explorados. Inicialmente, o nó de partida é colocado na fila.
Conjunto de Visitados: Um conjunto (visitados) é utilizado para garantir que o algoritmo não explore um nó mais de uma vez.
Dicionário de Predecessores: Um dicionário (predecessores) armazena o caminho percorrido, associando a cada nó (filme) o nó que o precedeu na busca. Isso será útil no backtracking para reconstruir o caminho final.
Exploração em Largura: O algoritmo retira um filme da fila e explora todos os seus vizinhos (filmes conectados a ele). Para cada vizinho não visitado, ele é adicionado à fila e marcado como visitado, com o filme atual sendo registrado como seu predecessor.
Backtracking: Se o filme de destino for alcançado, a função reconstruir_caminho utiliza o dicionário de predecessores para traçar o caminho do destino até o filme inicial. Caso contrário, a função retorna uma lista vazia.

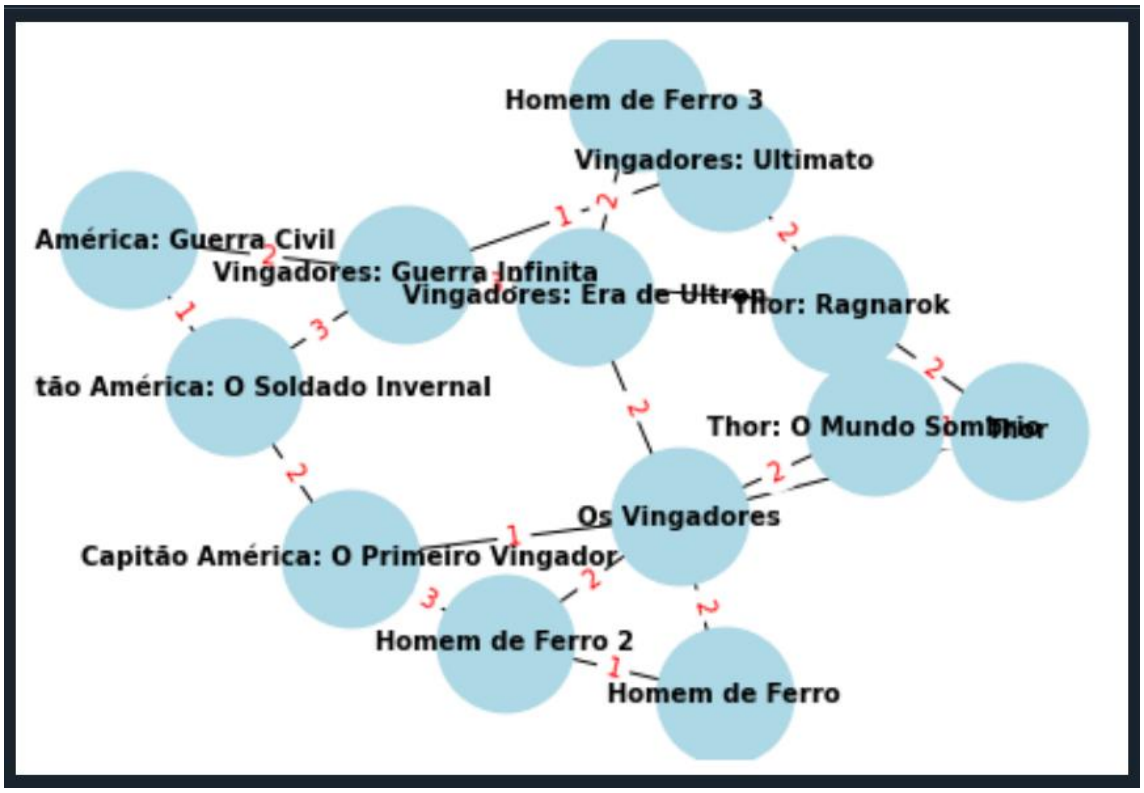
3. Visualização do Grafo

A função plotar_grafo usa a biblioteca networkx para criar uma representação gráfica do grafo. Cada filme é um nó, e as conexões entre eles (arestas) são desenhadas, com o peso das arestas exibido. A visualização permite uma compreensão intuitiva das relações entre os filmes e a proximidade entre eles.

A biblioteca matplotlib é usada para renderizar o grafo, onde cada nó (filme) é exibido em um layout organizado, com as arestas rotuladas com seus pesos.


Homem de Ferro 3
Vingadores: Ultimato
Capitão América: Gue
Vingadores: Guerra Infinita
Thor: Ragnarok
Vingadores: Era de Ultron
Capitão América: O Soldado In
Thor
Thor: O Mundo Sombrio
Os Vingadores
Capitão América: O Primeiro Vingado
Homem de Ferro 2
Homem de Ferro


América: Guerra Civil
Capitão América: O Soldado Invernal
Capitão América: O Primeiro Vingador
Vingadores: Guerra Infinita
Homem de Ferro 2
Vingadores: Era de Ultron
Os Vingadores
Homem de Ferro 3
Thor: Ragnarok
Thor: O Mundo Sombrio
Thor



4. Interação com o Usuário

O código permite ao usuário interagir com o sistema, solicitando que ele insira o nome do filme inicial e o nome do filme de destino. Caso o usuário insira um filme inválido (que não esteja no dicionário), ele será solicitado a tentar novamente até fornecer uma entrada válida. Isso garante que a busca seja realizada corretamente.

```
In [11]: runfile('C:/Users/gulli/OneDrive/Área de Trabalho/IA/Filmes_BFS.py', wdir='C:/Users/gulli/OneDrive/Área de Trabalho/IA')
Digite o nome do filme de partida: Homem de Ferro
Digite o nome do filme de destino: Capitão América: Guerra Civil
O caminho de Homem de Ferro até Capitão América: Guerra Civil é: Homem de Ferro -> Homem de Ferro 2 -> Capitão América: O Primeiro
Vingador -> Capitão América: O Soldado Invernal -> Capitão América: Guerra Civil

In [12]: runfile('C:/Users/gulli/OneDrive/Área de Trabalho/IA/Filmes_BFS.py', wdir='C:/Users/gulli/OneDrive/Área de Trabalho/IA')
Digite o nome do filme de partida: Capitão América: O Primeiro Vingador
Digite o nome do filme de destino: Thor: Ragnarok
O caminho de Capitão América: O Primeiro Vingador até Thor: Ragnarok é: Capitão América: O Primeiro Vingador -> Os Vingadores ->
Thor -> Thor: Ragnarok

In [13]:
```

Conclusão

O código utiliza algoritmos de grafos para resolver o problema de encontrar o menor caminho entre dois filmes no MCU. A conclusão deste processo inclui:

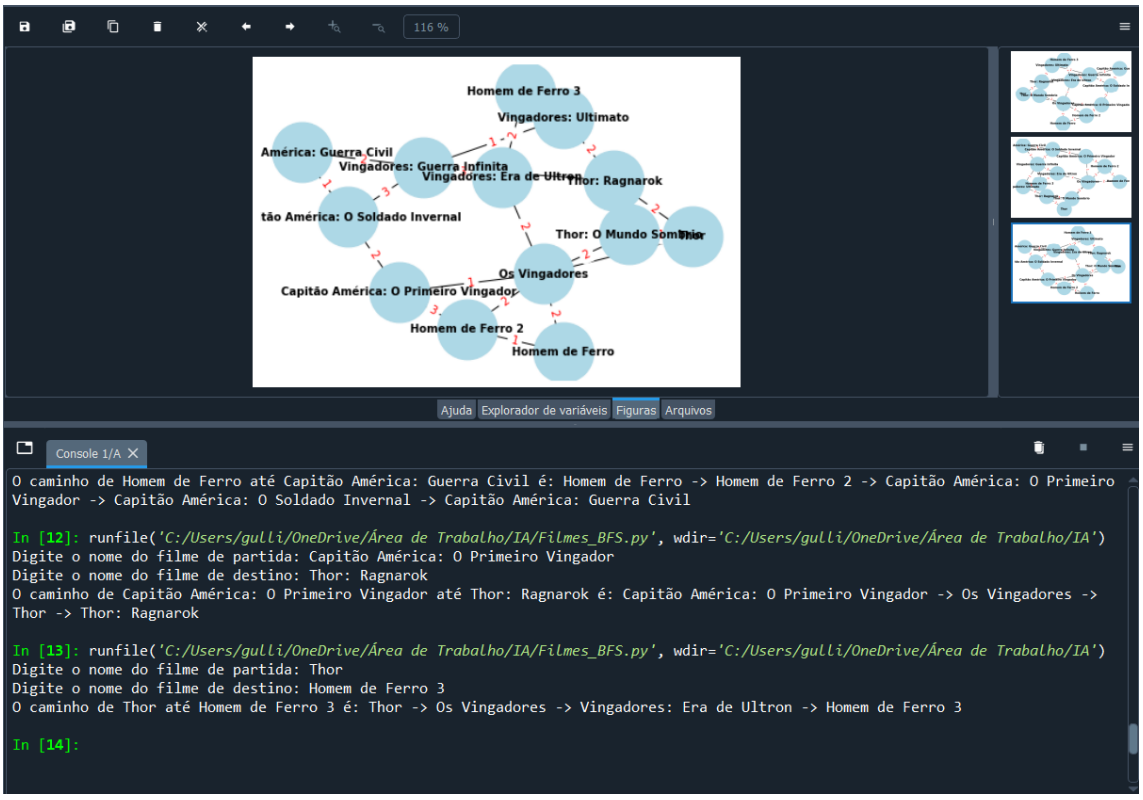
Eficiência do Algoritmo BFS: O uso do algoritmo BFS garante que o menor caminho (em termos de número de arestas) entre dois filmes seja encontrado, uma vez que o BFS explora todos os caminhos possíveis de maneira sistemática. O algoritmo é apropriado para esse tipo de problema, pois o número de nós (filmes) e arestas (conexões) é relativamente pequeno.

Backtracking para Reconstruir Caminho: A técnica de backtracking permite reconstruir o caminho após a conclusão da busca, garantindo que o usuário possa ver a sequência exata de filmes que leva do ponto inicial ao ponto de destino.

Visualização para Intuição: A visualização gráfica do grafo dos filmes facilita a compreensão das inter-relações entre eles. Isso pode ser útil, por exemplo, para analisar a narrativa do MCU ou ver como os filmes estão conectados de acordo com suas aparições conjuntas.

Flexibilidade: O código é flexível o suficiente para ser adaptado a outros domínios que envolvam grafos e busca de caminhos, como redes sociais, mapas de rotas, etc. O dicionário lista_de_filmes pode ser facilmente expandido com mais filmes ou outros tipos de nós, e o código continuará funcionando da mesma maneira.

Em resumo, o código implementa uma solução eficaz para encontrar e visualizar o menor caminho entre dois filmes no universo do MCU, utilizando técnicas de grafos e algoritmos clássicos, como BFS e backtracking.



Console 1/A

```
O caminho de Homem de Ferro até Capitão América: Guerra Civil é: Homem de Ferro -> Homem de Ferro 2 -> Capitão América: O Primeiro
Vingador -> Capitão América: O Soldado Invernal -> Capitão América: Guerra Civil

In [12]: runfile('C:/Users/gulli/OneDrive/Área de Trabalho/IA/Filmes_BFS.py', wdir='C:/Users/gulli/OneDrive/Área de Trabalho/IA')
Digite o nome do filme de partida: Capitão América: O Primeiro Vingador
Digite o nome do filme de destino: Thor: Ragnarok
O caminho de Capitão América: O Primeiro Vingador até Thor: Ragnarok é: Capitão América: O Primeiro Vingador -> Os Vingadores ->
Thor -> Thor: Ragnarok

In [13]: runfile('C:/Users/gulli/OneDrive/Área de Trabalho/IA/Filmes_BFS.py', wdir='C:/Users/gulli/OneDrive/Área de Trabalho/IA')
Digite o nome do filme de partida: Thor
Digite o nome do filme de destino: Homem de Ferro 3
O caminho de Thor até Homem de Ferro 3 é: Thor -> Os Vingadores -> Vingadores: Era de Ultron -> Homem de Ferro 3

In [14]:
```