

Universidade Federal de Pernambuco
Centro de Informática - CIn

Disciplina: Algoritmos e Estruturas de Dados

Docente: Sérgio Queiroz

Discente:

Hítalo Nascimento

Ingrid Freire

Relatório do projeto: Grupo #1.1

Recife, 16 de abril de 2023

Contexto do problema

Uma empresa de rede social deseja fornecer a seus usuários uma maneira rápida e eficiente de se conectarem entre si, com o menor número possível de saltos entre eles. Para resolver esse problema, a empresa decidiu utilizar o algoritmo de Dijkstra, que é um algoritmo de busca em grafos que determina o menor caminho entre um nó de origem e todos os outros nós de um grafo ponderado, no qual cada aresta possui um peso.

A base de dados escolhida é baseada nos usuários da plataforma Last.fm Asia, coletada em março de 2020. Possui 7.624 nós e 27.806 arestas, e pode ser acessada pelo link <http://snap.stanford.edu/data/feather-lastfm-social.html>. Neste problema, o objetivo é saber o menor caminho para conectar 2 usuários utilizando o algoritmo Dijkstra.

Implementação

Algoritmo utilizado: Dijkstra

Desenvolvimento.

O projeto seguiu uma abordagem modular, onde cada módulo é responsável por uma parte específica do programa. Essa abordagem foi tomada para reaproveitamento de código entre o Jupyter Notebook e o arquivo main.py. O diretório *data_base* contém os dados necessários para a aplicação do algoritmo de Dijkstra, enquanto no diretório *modules*, o módulo graph.py define a estrutura de dados utilizada pelo algoritmo. O módulo dijkstra.py contém a implementação do algoritmo de Dijkstra propriamente dito, e o módulo data.py contém funções auxiliares para carregar e manipular os dados. O arquivo main.py é responsável por executar o algoritmo de Dijkstra com os dados fornecidos por meio de uma interface gráfica. O arquivo main.ipynb é um notebook Jupyter que contém um exemplo de uso do programa. O repositório pode ser acessado pelo link <https://github.com/HitaloNasc/dijkstra-algorithm.git>.

Estrutura de diretórios

```
dijkstra-algorithm
├── README.md
├── data_base
│   ├── README.txt
│   ├── lastfm_asia_edges.csv
│   ├── lastfm_asia_features.json
│   └── lastfm_asia_target.csv
├── main.ipynb
├── main.py
└── modules
    ├── __init__.py
    ├── data.py
    ├── dijkstra.py
    ├── graph.py
    └── setup.py
```

Bibliotecas utilizadas.

Para leitura dos dados utilizados na base de dados no formato csv foi usada a biblioteca csv. Além disso, para uma melhor organização dos dados lidos a biblioteca *typing*

foi aplicada.

O algoritmo de Dijkstra foi construído usando Python puro, seguindo as regras do exercício.

Para o construir o grafo usado pelo Dijkstra apenas Python puro foi utilizado. Para exibir o grafo no formato de gráfico foi utilizado *networkx* e *matplotlib*.

Para a interface gráfica do projeto foi usado *tkinter*.

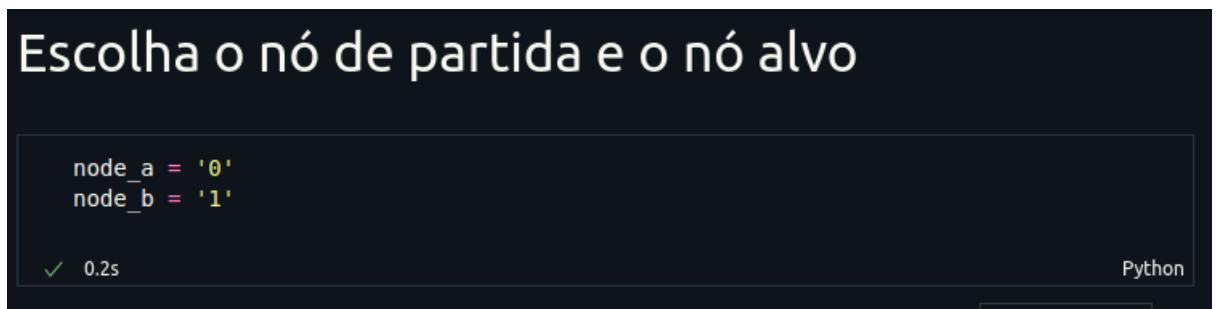
Para automatizar a instalação das dependências foi usada a biblioteca *subprocess*.

Conclusão

O programa lê a base de dados no diretório *data_base* e pode ser executado de duas maneiras:

A partir do Jupyter Notebook

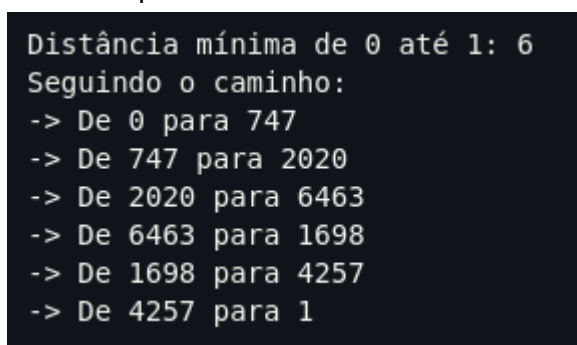
1. Abra o notebook
2. Escolha o nó de partida e o nó de chegada atribuindo as variáveis `node_a` e `node_b`



The screenshot shows a Jupyter Notebook interface with a dark theme. The title of the cell is "Escolha o nó de partida e o nó alvo". The code cell contains two lines of Python code: `node_a = '0'` and `node_b = '1'`. Below the code, there is a green checkmark icon, the text "0.2s", and the word "Python" in the bottom right corner.

```
node_a = '0'
node_b = '1'
```

3. Execute o notebook
4. Acompanhe os resultados
- 4.1 Rota percorrida



The screenshot shows a terminal window with a dark background. The output text is as follows:

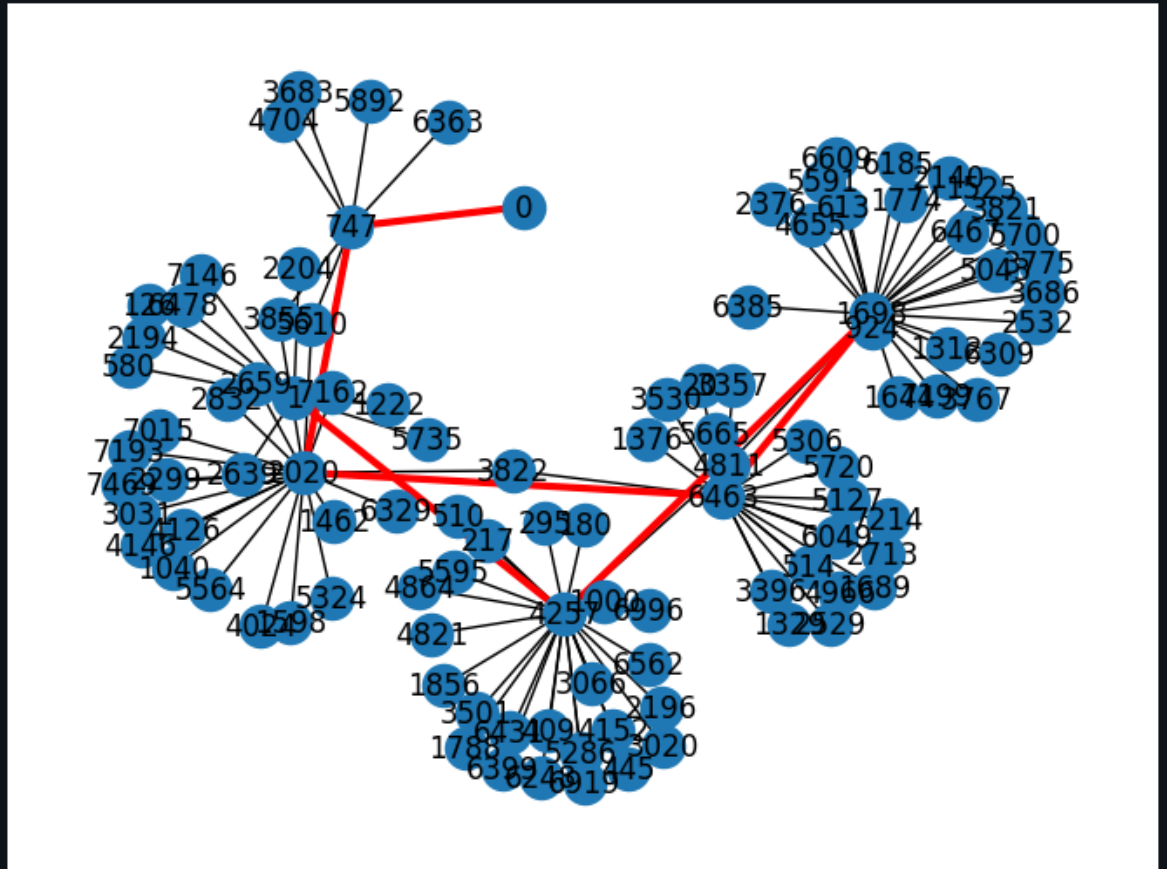
```
Distância mínima de 0 até 1: 6
Seguindo o caminho:
-> De 0 para 747
-> De 747 para 2020
-> De 2020 para 6463
-> De 6463 para 1698
-> De 1698 para 4257
-> De 4257 para 1
```

- 4.2 Grafo

Ilustrando o menor caminho

```
graph.show_route(graph_data, route)
```

✓ 0.9s



A partir da interface gráfica

1. Execute o arquivo main.py

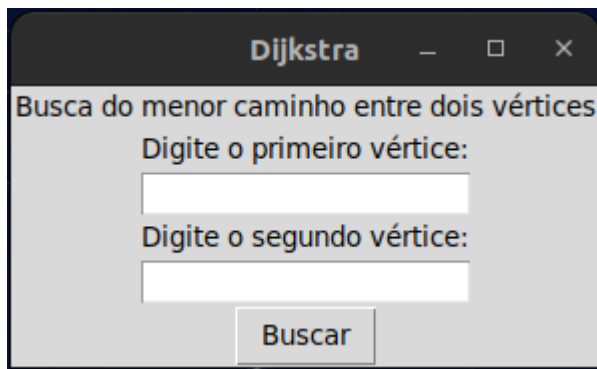
Executando via terminal. Na pasta do projeto execute:

```
$ python3 main.py
```

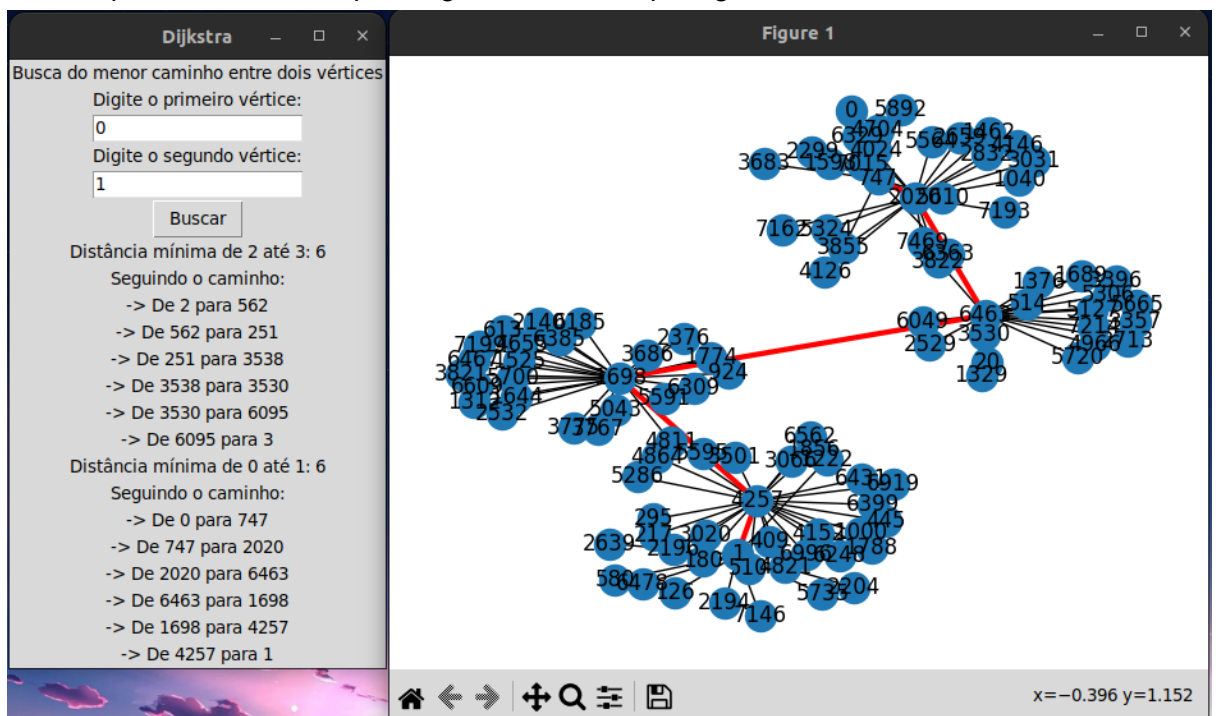
Executando via Code Runner

Abra o arquivo main.py e clique em executar (botão Code Runner) ou pelo atalho (ctrl + Alt + N)

2. Selecione o valor dos nós na interface e clique em "Buscar"



3. Acompanhe o resultado pelo log na interface e pelo grafo



Referências

Base de dados. Acesso em: <<http://snap.stanford.edu/data/feather-lastfm-social.html>>.