SERVIÇO PÚBLICO FEDERAL · MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE VIÇOSA · UFV
CAMPUS FLORESTAL

Trabalho prático 2 - Teoria e Modelo de Grafos

Bruno Vicentini Ribeiro - 5907
Erich Pinheiro Amaral - 5915
Pedro Paulo Paz do Nascimento - 5937
Vitor Mathias Andrade - 5901

Florestal - MG
2025

SUMÁRIO

INTRODUÇÃO.....3
DESENVOLVIMENTO..... 3
RESULTADOS.....5
CONCLUSÃO.....6
REFERÊNCIAS..... 6

INTRODUÇÃO

Este trabalho tem como objetivo resolver o problema de alocação de horários em uma universidade, onde disciplinas que compartilham o mesmo professor ou possuem alunos em comum não podem ser ministradas no mesmo horário. O problema foi modelado como um grafo de conflitos, no qual cada vértice representa uma disciplina e as arestas representam conflitos de horário. A solução consiste em aplicar algoritmos de coloração de grafos para atribuir horários (cores) às disciplinas, minimizando o número total de horários necessários.

A implementação foi realizada em Python, utilizando as bibliotecas NetworkX para manipulação de grafos e GCol para coloração. O sistema é capaz de ler um arquivo CSV com os conflitos, construir o grafo, executar múltiplos algoritmos de coloração e comparar os resultados.

DESENVOLVIMENTO

- ESTRUTURA DO PROGRAMA

O programa foi organizado em dois arquivos principais: leitura.py, responsável pela leitura do arquivo CSV e construção do grafo e main.py, que executa os algoritmos de coloração, compara os resultados e exibe o melhor. Além disso, o programa tem as pastas com as bibliotecas utilizadas: NetworkX e GCol.

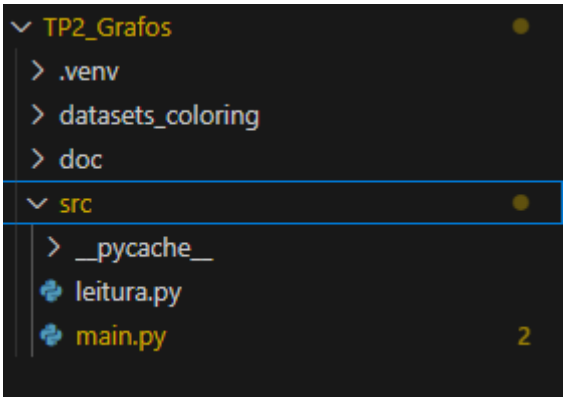

Figura 1: Repositório do projeto

- INSTALAÇÃO DAS DEPENDÊNCIAS

Para que o projeto possa ser executado corretamente, é necessário ter o Python instalado e configurar o ambiente com as bibliotecas que foram utilizadas. As principais dependências do projeto são:

- pandas: Utilizada para a leitura e manipulação de dados dos arquivos `.csv`.

- NetworkX: Usada para a criação, manipulação e estudo da estrutura do grafo de conflitos.
- GCol: Biblioteca principal que fornece os algoritmos de coloração de grafos.

A forma mais simples de instalar essas bibliotecas é utilizando o gerenciador de pacotes `pip` do Python.

Instalação via pip

Recomenda-se o uso de um ambiente virtual (como o `venv`, que pode ser visto na estrutura do projeto) para isolar as dependências.

Com seu ambiente virtual ativado, execute o seguinte comando no terminal para instalar todas as bibliotecas necessárias de uma só vez:

Bash

```
pip install pandas networkx gcol
```

Este comando fará o download e a instalação das três bibliotecas e suas respectivas dependências, deixando o ambiente pronto para a execução do programa.

- MODELAGEM DO GRAFO DE CONFLITOS

O grafo foi construído a partir de um arquivo CSV contendo pares de disciplinas em conflito. Cada linha representa uma restrição de horário entre duas disciplinas. A biblioteca NetworkX foi utilizada para criar o grafo não direcionado a partir desses dados.

- IMPLEMENTAÇÃO DA COLORAÇÃO COM GCOL

Foram testados quatro algoritmos de coloração disponíveis na biblioteca GCOL: DSATUR, RLF, Random e Welsh-Powell.

Cada algoritmo foi executado sobre o mesmo grafo e os resultados comparados com base no número de cores utilizadas e no tempo de execução.

- REQUISITOS E EXECUÇÃO

Para a correta execução do projeto, é necessária a instalação de três bibliotecas principais do Python:

- pandas: Utilizada para a leitura e manipulação inicial dos dados a partir dos arquivos .csv.
- NetworkX: Empregada para a modelagem e construção do grafo de conflitos.
- GCol: Biblioteca central para a aplicação dos algoritmos de coloração.

O programa principal, main.py, é projetado para ser executado através do terminal. Ele recebe como argumento obrigatório o caminho para o dataset .csv que contém as arestas (conflitos) do grafo.

O comando de execução segue o formato: python3 src/main.py [caminho_para_o_dataset] Por exemplo, para processar o arquivo grande.csv localizado na pasta datasets_coloring, o comando seria:

Bash:

```
python src/main.py datasets_coloring/grande.csv
```

RESULTADOS

O script main.py foi desenvolvido para executar e comparar os quatro algoritmos de coloração (DSATUR, RLF, Random e Welsh-Powell) disponibilizados pela biblioteca GCol. A comparação avalia dois critérios principais: o número de cores (horários) utilizado e o tempo de execução. O "Melhor Resultado" é definido como o algoritmo que utiliza o menor número de cores e, em caso de empate, o que executa no menor tempo.

Como exemplo, a Figura 2 abaixo exibe a saída do terminal referente à execução do programa com o dataset pequeno.csv.

```
Grafo de conflitos montado com sucesso.
  - 5 disciplinas (vértices)
  - 5 conflitos (arestas)
----Comparação de Algoritmos----

   Algoritmo N° de Cores    Tempo
welsh_powell           2 0.000015
      random           2 0.000030
         rlf           2 0.000057
      dsatur           2 0.000268

--- Melhor Resultado ---
Algoritmo: welsh_powell
Número de cores (Horários): 2
Tempo de execução aproximado: 0.000015 segundos


  - C: Horário 0
  - A: Horário 1
  - B: Horário 0
  - D: Horário 1
  - E: Horário 1
```

Figura 2: Resultado da execução para o dataset pequeno.csv.

Analisando a Figura 2, nota-se que o grafo de conflitos foi montado com 5 disciplinas (vértices) e 5 conflitos (arestas). Todos os algoritmos testados encontraram o número cromático de 2, ou seja, são necessários no mínimo 2 horários distintos.

O algoritmo welsh_powell foi o mais eficiente em termos de velocidade para este conjunto de dados, com um tempo aproximado de 0.000015 segundos. A alocação de horários sugerida foi:

- **Horário 0:** Disciplinas C e B
- **Horário 1:** Disciplinas A, D e E

CONCLUSÃO

Este trabalho demonstrou com sucesso a aplicação prática da teoria dos grafos para a resolução de um problema complexo de alocação de horários em uma universidade. O problema foi eficientemente modelado como um grafo de conflitos , onde a coloração de vértices corresponde diretamente à atribuição de horários.

A utilização das bibliotecas Python, especialmente NetworkX para a estruturação dos dados e GCol para as heurísticas de coloração, provou ser uma abordagem robusta. O sistema desenvolvido é capaz de ler diferentes conjuntos de conflitos , processá-los e comparar múltiplos algoritmos , identificando a solução mais otimizada (menor número de horários) e o algoritmo mais performático para cada cenário.

O projeto cumpre todos os objetivos propostos, entregando uma ferramenta funcional que minimiza o número de horários necessários e, consequentemente, otimiza a alocação de recursos da instituição.

REFERÊNCIAS

[1] GCol: A Library for Graph Coloring “https://gcol.readthedocs.io/en/latest/”
[2] Documentação do NetworkX 3.5
“https://networkx.org/documentation/stable/tutorial.html”
[3] Python Data Analysis Library
“http://pandas.pydata.org/docs/getting_started/index.html#getting-started”