

Integrando MST no Planejamento de Redes Ferroviárias/Metroviárias

José Marconi de Almeida Júnior

Engenharia de Computação

CEFET - Campus V

Divinópolis, MG

jmarconiadm@outlook.com

Abstract—Este trabalho propõe uma abordagem integrada no planejamento de um sistema ferroviário/metroviário nacional no Brasil, excluindo as atuais existentes no país. O foco principal é a identificação das rotas mais curtas e eficientes entre as capitais, visando estabelecer uma malha ferroviária/metroviária que conecte todo o país a partir de um ponto estratégico em cada Estado para minimizar custos de construção e otimizar a distribuição de recurso. Utilizando as técnicas de Caminho Mínimo como suporte para finalizar com a Árvore de Abrangência Mínima (MST) a partir do algoritmo de Kruskal, implementados com o Python 3 e com grafos modelados a partir da utilização do software Gephi. A análise prova a eficiência do algoritmo de Kruskal para essa utilização.

Palavras-Chave—Malha Ferroviária/Metroviária, árvore geradora mínima(MST), Kruskal, Teoria dos Grafos.

I. INTRODUÇÃO

A eficiência logística desempenha um papel crucial no desenvolvimento e no progresso socioeconômico de uma nação. No contexto específico de sistemas ferroviários e metroviários, a otimização do planejamento desempenha um papel vital para garantir conectividade eficiente, minimização de distâncias entre centros urbanos e a alocação eficaz de recursos. Este trabalho visa abordar de maneira integrada o desafio complexo do planejamento de redes ferroviárias e metroviárias nacionais, com um enfoque especial no cenário brasileiro.

O Brasil, um país vasto em território, enfrenta desafios logísticos significativos ao buscar estabelecer uma infraestrutura ferroviária e metroviária abrangente. Há apenas algumas malhas ferroviárias que são concentradas principalmente nas regiões Sul e Sudeste do país, algumas poucas na Nordeste e Centro Oeste, ou seja, não existe uma malha que conecte todos os Estados. Nesse contexto, este projeto propõe uma abordagem para o planejamento de uma malha ferroviária/metroviária nacional, excluindo as infraestruturas existentes, com o objetivo principal de identificar as rotas mais curtas e eficientes entre as capitais estaduais. A implementação de uma malha ferroviária/metroviária eficiente, conectando estrategicamente todo o país a partir de pontos nodais em cada estado, não apenas otimiza a conectividade, mas também oferece oportunidades substanciais para a racionalização da alocação de recursos e a redução de custos de construção.

O foco central desta pesquisa é empregar técnicas avançadas de otimização, notadamente os algoritmos de Caminho

Mínimo e Árvore de Abrangência Mínima (MST), utilizando a linguagem de programação Python 3 (versão 3.11.6). A modelagem dos grafos, essenciais para a representação eficaz das redes ferroviárias/metroviárias é realizada com o auxílio do software Gephi. A análise resultante visa não apenas minimizar as distâncias entre as capitais, mas também proporcionar uma conectividade eficiente que reflita diretamente na otimização da alocação de recursos e na redução dos custos associados à construção de infraestruturas ferroviárias e metroviárias, buscando um meio eficiente de transporte a longas distâncias de mercadorias e pessoas.

Por meio da integração de técnicas avançadas de otimização, este trabalho busca não apenas traçar rotas eficientes, mas também estabelecer as bases para um planejamento logístico sustentável e economicamente viável, que atenda às demandas presentes e futuras da infraestrutura de transporte do Brasil.

II. TRABALHOS CORRELATOS

Ao explorar a literatura existente, ficou evidente que não havia um estudo abrangente que tratasse da construção de malhas para a conexão de todo o território brasileiro, utilizando a teoria dos grafos de maneira específica à nossa abordagem. Esta lacuna na pesquisa motivou a elaboração deste projeto cuja proposta foi moldada com base em uma análise de dois trabalhos.

O primeiro trabalho, intitulado "ANÁLISE LOGÍSTICA DA MATRIZ FERROVIÁRIA NACIONAL ATRAVÉS DE PESQUISA OPERACIONAL," apresentado por K. J. Faria, J. V. S. M. Catarino, D. F. Vicentini na Universidade Federal do Paraná (UFPR), durante o 1º SIMPÓSIO DE TRANSPORTES DO PARANÁ (STPR), realizado entre 21 e 23 de março de 2018, serviu como uma fonte de inspiração significativa para nosso projeto. Embora o referido trabalho tenha abordado a análise das malhas ferroviárias existentes, concentrou-se principalmente na aplicação da teoria dos grafos para modelagem visual. Em contraste, nossa pesquisa busca ir além, empregando a teoria dos grafos não apenas como uma ferramenta de representação, mas como a base fundamental para a construção de malhas ferroviárias que conectem eficientemente todo o território brasileiro.

O segundo trabalho, intitulado "Implementação de um algoritmo para solução do caminho de custo mínimo na logística

aplicada ao transporte rodoviário de soja do estado de Mato Grosso,” desenvolvido por Renan Alves do Nascimento mestre em Engenharia de Produção e de Manufatura na Universidade Estadual de Campinas (UNICAMP), publicado em 20 de junho de 2020, desempenhou um papel fundamental na orientação de nossa pesquisa. Embora o trabalho em questão tenha se concentrado na aplicação do algoritmo de caminho mínimo de Dijkstra para a geração de uma árvore geradora mínima no contexto do transporte de soja, nossa abordagem é distinta ao aplicar essas técnicas em uma escala nacional, visando a construção de uma malha ferroviária e metroviária abrangente para conectar diversas regiões do Brasil. Ao diferenciar nosso escopo para além do transporte rodoviário específico de uma commodity, nossa pesquisa procura aplicar e estender esses métodos para a infraestrutura de transporte em larga escala.

III. METODOLOGIA

A teoria dos grafos é um ramo da matemática que estuda a representação e análise de relações entre objetos. Em sua essência, um grafo é uma estrutura composta por vértices (ou nós) e arestas (ou arcos) que conectam esses vértices. Essa representação visual permite modelar uma variedade de problemas do mundo real, um deles é a infraestruturas des transportes que é o ponto desse trabalho. E usufruiremos de dois algoritmos principais relacionados a grafos, sendo eles, o algoritmo de Dijkstra e o algoritmo de Kruskal.

A linguagem utilizada para a implementação foi o Python 3 na versão 3.11.6. As bibliotecas utilizadas foram a *pandas* utilizada principalmente para manipulação e análise de dados tabulares, nesse caso para manipulação dos arquivos csv, e a *networkx* para a criação, manipulação e análise de redes complexas e grafos, nesse caso utilizando um algoritmo variante de Dijkstra como suporte e Kruskal para a Árvore Geradora Mínima(MST).

Foi utilizado também o software Gephi para visualização e manipulação dos grafos a partir dos arquivos CSV. Com ele é possível modelar os grafos e manipular seus dados diretamente através de seu layout.

O ponto principal do trabalho é a utilização da MST, o algoritmo de Dijkstra na implementação tem uma funcionalidade suporte, e o Kruskal atuando os resultados.

A. Dijkstra

O algoritmo de Dijkstra é um algoritmo de caminho mínimo que encontra o caminho mais curto entre dois nós em um grafo ponderado com arestas não negativas. Ele foi concebido pelo cientista da computação holandês Edsger Dijkstra em 1956. Seu formato clássico de implementação está representado no algoritmo 1.

O algoritmo trabalha com vértices em que já é conhecida a sua menor distância até a raiz e também aqueles em que sua distância é ainda provisória. Ele também utiliza de estruturas auxiliares, tais como $\pi[u]$ armazenando o pai de cada vértice u encontrado, $d[u]$ que a armazena a distância de origem até

u , Q armazena os nós com distância provisória e S armazena os nós com distância definitiva.

Algorithm 1 Algoritmo de Dijkstra

```

1: function DIJKSTRA( $G, w, s$ )
2:   for each  $v \in V[G]$  do
3:      $d[v] \leftarrow \infty$ 
4:      $\pi[v] \leftarrow \text{NULL}$ 
5:   end for
6:    $S \leftarrow \{\}$ 
7:    $Q \leftarrow V[G]$ 
8:   while  $Q \neq \emptyset$  do
9:      $u \leftarrow \text{ExtractMin}(Q)$ 
10:     $S \leftarrow S \cup \{u\}$ 
11:    for each  $v \in \text{Adj}[u]$  do
12:      if  $d[v] > d[u] + w(u, v)$  then
13:         $d[v] \leftarrow d[u] + w(u, v)$ 
14:         $\pi[v] \leftarrow u$ 

```

Na implementação realizada para a pesquisa, é utilizada uma versão de Dijkstra da biblioteca *networkx* a partir da função `all_pairs_dijkstra_path_length()`, que irá retornar um dicionário com todos os comprimentos de caminho mais curto entre todos os nós em um grafo ponderado.

B. Kruskal

O algoritmo de Kruskal é um algoritmo guloso que encontra uma árvore geradora mínima (Minimum Spanning Tree - MST) em um grafo ponderado e conexo. Uma árvore geradora mínima é uma árvore que abrange todos os vértices do grafo, sem formar ciclos, e cuja soma dos pesos das arestas é minimizada. Seu algoritmo clássico é representado pelo algoritmo 2.

Algorithm 2 Algoritmo de Kruskal (MST)

```

1: function MSTKRUSKAL( $G, w$ )
2:    $A \leftarrow \text{NULL}$ 
3:   for each  $v \in V[G]$  do
4:     CriarConjunto( $v$ )
5:   end for
6:    $A' \leftarrow \text{SortEdges}(E, w)$ 
7:   for each  $(u, v) \in A'$  do
8:     if FindSet( $u$ )  $\neq$  FindSet( $v$ ) then
9:        $A \leftarrow A \cup \{u, v\}$ 
10:      Union( $u, v$ )
11:     end if
12:   end for
13:   Return  $A$ 

```

Na implementação realizada para a pesquisa, foi utilizado a função `minimum_spanning_edges()` presente na biblioteca *networkx* para a geração da nossa MST utilizando Kruskal.

Foi utilizada uma malha para servir como parâmetro, representada na figura 1. A princípio o objetivo era atingir algo

mais otimizado em vista que ela contém um número maior de arestas que os resultados encontrados pela MST, que contém um número máximo de arestas igual ao número de nós menos um.

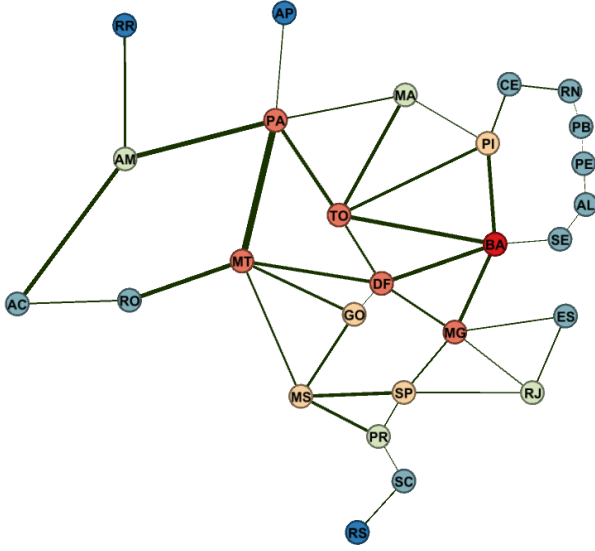


Fig. 1. Grafo modelado a mão no Gephi a partir do arquivo Ideal.csv, não foi utilizado nenhum algoritmo para formular esse arquivo. Quanto maior o grau das arestas (referente as distâncias), mais espessa ela é. Quanto maior o grau dos nós, mais próximo do vermelho, quanto menor, mais próximo do azul. Número de nós: 27, Número de arestas: 40

C. Implementação Prática

O primeiro passo para a realização do processo, foi adquirir os dados que serão utilizados. Foi utilizado um arquivo em csv chamado *capitais* contendo a UF de cada estado e sua latitude e longitude, representadas na Tabela 1, sendo centradas em suas capitais a partir do google maps.

É realizada a leitura do arquivo utilizando das funções presentes na biblioteca *pandas*. Em seguida criamos o grafo a partir da biblioteca *networkx* com os valores lidos do CSV. Em seguida é realizado cálculo das distâncias euclidianas e adicionando arestas ponderadas ao grafo.

O cálculo das distâncias euclidianas é dado pela equação 1, em que temos latitude como x e longitude como y . É utilizado uma implementação simples, de dois *For* para realização dos cálculos e adicionamos os nós e os resultados para montarmos um grafo.

$$D = \sqrt{(x_f - x_i)^2 + (y_f - y_i)^2} \quad (1)$$

TABLE I
UF COORDENADAS

UF	Latitude	Longitude
AC	-9.971450	-67.809811
AL	-9.643221	-35.734441
AM	-3.111853	-60.015173
AP	0.019375	-51.060472
BA	-12.965910	-38.502327
CE	-3.756945	-38.535504
DF	-15.792622	-47.881571
ES	-20.293941	-40.306819
GO	-16.684349	-49.246936
MA	-2.550855	-44.259499
MT	-15.600334	-56.090693
MS	-20.460495	-54.622834
MG	-19.911736	-43.934528
PA	-1.461134	-48.478110
PB	-7.126162	-34.868555
PR	-25.441091	-49.271615
PE	-8.064941	-34.887384
PI	-5.068724	-42.802730
RJ	-22.907855	-43.227983
RN	-5.815283	-35.206798
RO	-8.760446	-63.898510
RS	-30.054104	-51.182475
RR	2.823991	-60.676673
SC	-27.595777	-48.579419
SE	-10.929907	-37.071316
SP	-23.562735	-46.639596
TO	-10.246367	-48.324149

Após todos os cálculos, utilizamos a função `all_pairs_dijkstra_path_length()` para percorrer o grafo e retornar um dicionário com as distâncias mínimas de cada vértice até todos os outros vértices através de um algoritmo que utiliza da estratégia de Dijkstra, que será salvo em um arquivo CSV denominado de *Dijkstra.csv*.

A partir desse novo arquivo, é realizado a modelagem visual desse grafo no software Gephi, importando no formato de Matriz, sendo ele também dirigido. Modelando assim, um grafo de 27 nós e 702 aresta. Em seguida, a partir dos dados disponíveis na aba "Laboratório de dados", foi realizada duas podas manuais nas arestas.

A condição para a poda foi de manter as duas menores distâncias de cada UF para o primeiro teste e as quatro menores para o segundo teste. Ordenando a aba "Origem" em ordem alfabética e a aba "Weight" dos menores para os maiores e sem repetir as mesmas UF, por exemplo tendo como origem Acre(AC) e destino Amazonas(AM) é possível que ocorresse tendo como origem Amazonas(AM) e destino (Acre), o que seria uma repetição, nesse caso de repetição, foram descartadas, e escolhidas a próxima com menor distância. Ao finalizar a poda manual, é exportado o novo arquivo CSV como *CaminhoMinimo2.csv* e *CaminhiMinimo4.csv*.

É realizada então a leitura do novo arquivo e a criação de um novo grafo ponderado direcionado. O grafo então é convertido em não direcionado e é retirado dele uma árvore geradora mínima utilizando a função `minimum_spanning_edges()`, que aplica um algoritmo de Kruskal. A árvore é então salva em um arquivo CSV

chamados MST.csv(para a poda de dois) e MST2.csv(para a poda de 4).

Por fim, é utilizado novamente o Gephi para modelar visualmente a MST obtida como o resultado final. Dessa vez importando o arquivo como "Edges-table" e também não dirigido. Retornando um grafo não dirigido de 27 nós e 26 arestas.

IV. RESULTADOS

A implementação prática da metodologia proposta revelou resultados interessantes para a construção de uma malha conectando todos os estados do país. Os resultados em si foram satisfatórios para testar a eficiência do MST para a possível modelagem de uma malha real.

O código implementado para o projeto se encontra no link do github <https://github.com/josemarconi/Integrando-MST-no-Planejamento-de-Redes-Ferrovias-Ferrovias>. git.

A modelagem do grafo obtido após a aplicação de Djiskra para encontrar os caminhos mínimos está disposto na figura 2, representando todas as UF e todas as suas ligações coma as demais, o grafo em si não é tão nítido pela quantidade de arestas.

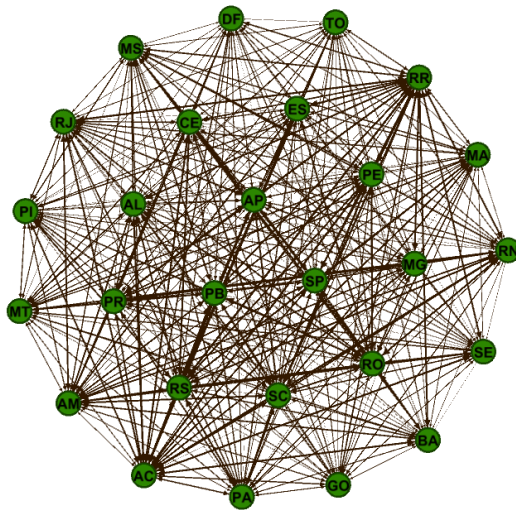


Fig. 2. Grafo modelado a partir do arquivo Dijkstra.csv, utilizando da função derivada de Dijkstra para o cálculo do menor caminho de cada vértice até os outros. O grafo apresenta todas as UF e é um grafo direcionado, quanto mais forte o tom da aresta, maior o peso dela. Número de nós: 27, Número de arestas: 702

Após a realização das podas, a figura 3 representa o grafo de duas distâncias e a figura 4 para a de quatro distâncias, eles representam exatamente a mesma coisa que o grafo da figura 1, porém com as podas realizadas, logo suas conexões se encontram levemente mais nítidas.

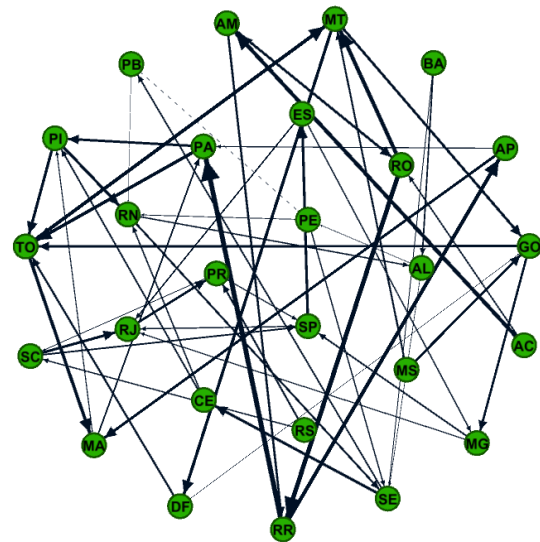


Fig. 3. Grafo modelado a partir do arquivo CaminhoMínimo2.csv para visualização do grafo após a poda com a condição de duas menores distâncias sem repetição. É um grafo direcionado, quanto mais forte o tom da aresta, maior o peso dela. Número de nós: 27, Número de arestas: 54

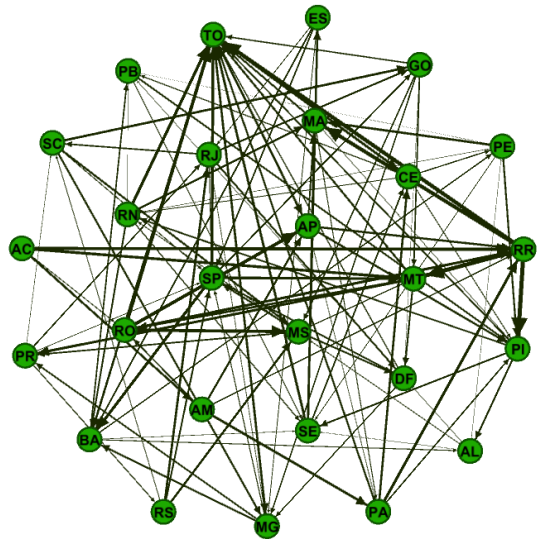
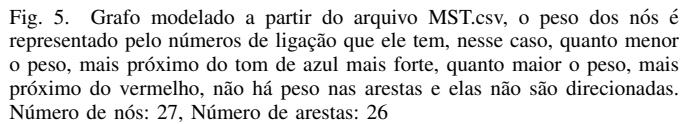


Fig. 4. Grafo modelado a partir do arquivo CaminhoMínimo4.csv para visualização do grafo após a poda com a condição de quatro menores distâncias sem repetição. É um grafo direcionado, quanto mais forte o tom da aresta, maior o peso dela. Número de nós: 27, Número de arestas: 108

As figuras 5 e 6 representam respectivamente as árvores geradoras mínimas referentes as as podas de duas distâncias e quatro distâncias, que utilizamos como resultado final da pesquisa.

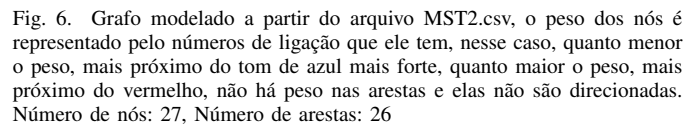
A análise do grafo modelado de duas distâncias nos permite tirar algumas informações importantes. Temos como principais conectores do país o estado do Mato Grosso(MT) que se tornou o principal ponto de ligação do Sul e Sudeste às

Porém, há um enorme problema nessa malha. Para ir do Sul, ou Sudeste para o Nordeste, e vice versa, é necessário dar uma volta completa pelo Norte, pois não há uma ligação entre Sudeste e Nordeste, dessa forma, a concetividade não se torna eficiente para esse caso, oque não pode acontecer.



A segunda malha apresentou um resultado um pouco mais eficiente que a primeira, entretanto, ainda há alguns sério problemas nela. Com o fato de que, para chegar ao Estado do Alagoas(AL) e Pernambuco(PE) por exemplo, seria necessário passar pelo Norte, no sentido Mato Grosso(MT), Acre(AC) e Roraima(RR), sendo que o estado do Tocantins(TO) poderia sim realizar essa transição também.

ir para Tocantis(TO), oque claramente não é interessante para o problema.



Entretanto, nenhuma das malhas pode ser utilizado como um modelo razoável pelos motivos citados anteriormente em suas análises.

É possível concluir que os parâmetros de duas e quatro distâncias adotados em conjunto do algoritmo de Kruskal não se apresentou como uma ótima forma de formular a malha ferroviária/metroviária, sendo assim, não é recomendado utilizar as malhas encontradas nas figuras 5 e 6, pois possuem baixa eficiência. O algoritmo de Kruskal vai sempre buscar a menor soma dos pesos das arestas possível, então sempre será um número de vértices igual a 26, a busca por otimizar a figura 1, não se apresentou interessante com a utilização de Kruskal, por mais que tenha encontrado vértices chaves e ligações importantes nos casos apresentados.

É recomendado para trabalhos futuros, aplicar outros algoritmos como principais, como busca em largura (BSF), busca em profundidade (DSF), ou o próprio Dijkstra utilizando da estratégia de caminho mínimo para encontrar malhas mais eficientes ou mais próximas do exemplo adotado, além de

utilizar de parâmetros específicos, por exemplo economia, acrescentando mais cidades ao plano da malha, não só as capitais dos estados, também considerar possíveis obstáculos que não permitiriam a passagem ou construção de trilhos e reutilizar das malhas também já existentes.

REFERENCES

- [1] K. J. Faria, J. V. S. M. Catarino, e D. F. Vicentini, "Análise Logística da Matriz Ferroviária Nacional Através de Pesquisa Operacional", em *1º Simpósio de Transportes do Paraná*, 21-23 de março de 2018.
- [2] R. A. Nascimento, "Implementação de um Algoritmo para Solução do Caminho de Custo Mínimo na Logística Aplicada ao Transporte Rodoviário de Soja do Estado de Mato Grosso", *Brazilian Journal of Business*, págs. 3128-3141, Jun. 20, 2020.
- [3] `all_pairs_dijkstra_path_length()`. NetworkX: Network Analysis in Python. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.shortest_paths.weighted.all_pairs_dijkstra_path_length.html.
- [4] `minimum_spanning_edges()`. David Eppstein. NetworkX: Network Analysis in Python. 2006. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.tree.mst.minimum_spanning_edges.html
- [5] Gephi. Gephi Documentation. <https://docs.gephi.org/>.
- [6] T. Cormen, E. Leiserson, R. L. Rivest, C. Stein. "Algoritmos: Teoria e Prática". Editora Campus, Rio de Janeiro, 2002.