UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS DE DOIS VIZINHOS
CURSO DE ESPECIALIZAÇÃO EM CIÊNCIA DE DADOS

GUILHERME AUGUSTO BARRETA

MODELAGEM FILOGENÉTICA DAS CEPAS DE SARS-COV-2 VIA ÁRVORE DE EXTENSÃO MÍNIMA

TRABALHO DE CONCLUSÃO DE CURSO DE ESPECIALIZAÇÃO

DOIS VIZINHOS
2022

GUILHERME AUGUSTO BARRETA

MODELAGEM FILOGENÉTICA DAS CEPAS DE SARS-COV-2 VIA ÁRVORE DE EXTENSÃO MÍNIMA

Trabalho de Conclusão de Curso de Especialização apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Orientador: Prof. Dr. Dalcimar Casanova
Coorientador: Prof. Dr. Paulo Júnior Varela

DOIS VIZINHOS
2022

GUILHERME AUGUSTO BARRETA

MODELAGEM FILOGENÉTICA DAS CEPAS DE SARS-COV-2 VIA ÁRVORE DE EXTENSÃO MÍNIMA

Trabalho de Conclusão de Curso de Especialização apresentado ao Curso de Especialização em Ciência de Dados da Universidade Tecnológica Federal do Paraná, como requisito para a obtenção do título de Especialista em Ciência de Dados.

Data de aprovação: 28 de Abril de 2022

Dalcimar Casanova
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Pato Branco

Ives Renê Venturini Pola
Doutorado
Universidade Tecnológica Federal do Paraná - Câmpus Pato Branco

Yuri Kaszubowski Lopes
Doutorado
Universidade do Estado de Santa Catarina

DOIS VIZINHOS
2022

RESUMO

Desde a pandemia iniciada em Wuhan na China, em 12 de dezembro de 2019, surgiram muitas cepas do SARS-CoV-2. Na esteira da pandemia um esforço internacional para o sequenciamento genômico foi instalado, com o objetivo de monitorar a evolução do vírus e auxiliar na tomada de decisão. No entanto a análise de sequências genéticas não é trivial e requer algumas transformações a fim de facilitar a análise. A estratégia utilizada nesse trabalho é a construção da árvore filogenética e, para isso, é necessário abstrair o problema como um grafo e aplicar um algoritmo de árvore de extensão mínima. As arestas são resultantes das dissimilaridades entre sequências genéticas. Essa dissimilaridade é medida com a distância de Levenshtein. A complexidade dessa abordagem é $\mathcal{O}(n^2(\overline{k})^2)$, sendo n o número de sequências e k o tamanho médio das sequências. Para reduzir a complexidade desse algoritmo, foi proposto também um algoritmo heurístico. Dessa maneira a complexidade passa a ser $\mathcal{O}(n(\overline{f}(n)+\overline{r}(n))(\overline{k})^2)$, sendo f o número médio de folhas visitadas e r o número médio de nós visitados (com exceção das folhas) até encontra o ótimo local. Também é válido que $\overline{f}(n)+\overline{r}(n) \leq n$. Os resultados indicam que o tempo de execução relativo ao algoritmo determinístico tende a diminuir ao mesmo tempo em que a soma mínima de arestas apresenta um erro médio de 6%. Espera-se que, com os resultados apresentados, esse trabalho possa servir como base de compreensão e predição de mutações do coronavírus em trabalhos futuros.

BARRETA, Guilherme Augusto. Modelagem filogenética das cepas de SARS-COV-2 via árvore de extensão mínima. 2022. 33 f. Trabalho de Conclusão de Curso de Especialização – Curso de Especialização em Ciência de Dados, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2022.

Palavras-chave: Grafo. cepas. coronavírus. SARS-CoV-2. MST.

ABSTRACT

Since the pandemic that started in Wuhan, China, on December 12, 2019, many strains of SARS-CoV-2 appear. With that, monitoring was necessary. Fortunately, there is a international effort for genomic sequencing and makes these data publicly available. Data alone do not produce relevant information until some strategy is used to manipulate them. The strategy used in this work is the construction of the phylogenetic tree. For this, it is necessary to abstract the problem as a graph and apply a minimum spanning tree (MST). The edges are the result of dissimilarities between genetic sequences. This dissimilarity is measured with Levenshtein distance. The complexity of this approach is $\mathcal{O}(n^2(\overline{k})^2)$, where n is the number of sequences and k is the average size of the sequences. To reduce the complexity of this algorithm, a heuristic algorithm was also proposed. In this way the complexity becomes $\mathcal{O}(n(\overline{f}(n)+\overline{r}(n))(\overline{k})^2)$, where f is the average number of visited leaves and r is the average number of visited nodes (with the exception of leaves) until it finds the optimal location. It is also valid that $\overline{f}(n) + \overline{r}(n) \leq n$. The results indicate that the time of execution relative to the deterministic algorithm tends to decrease at the same time that the minimal sum of edges has an average error of 6%. It is expected that, with the results presented, this work can serve as a basis for understanding and predicting mutations in the coronavirus in future work.

BARRETA, Guilherme Augusto. Phylogenetic Tree modeling of SARS-COV-2 strains by Minimum Spanning Tree. 2022. 33 f. Trabalho de Conclusão de Curso de Especialização – Curso de Especialização em Ciência de Dados, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2022.

Keywords: Graph. Strains. SARS-CoV-2. MST.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

MST	Minimum Spanning Tree
NCBI	National Center for Biotechnology Information
RNA	Ácido ribonucleico

SUMÁRIO

1 INTRODUÇÃO

Em 12 de dezembro de 2019 surgiram casos de um novo coronavírus originado em Wuhan na China, causando problemas respiratórios em seus hospedeiros humanos, doença conhecida como Covid19 (ZHOU et al., 2020). Inicialmente atribuiu-se o nome de 2019-nCOV e depois foi renomeado para SARS-CoV-2. Ao mesmo tempo, dado a facilidade de propagação do vírus, houve surtos no mundo inteiro, ocasionando uma pandemia. Paralelamente, iniciou-se uma corrida para o desenvolvimento das vacinas (BARRETO, 2020).

Conforme o SARS-CoV-2 se espalhava, dadas as mutações inerentes nos processos biológicos, novas cepas se desenvolviam. Essas cepas apresentam um risco ainda maior, podendo tanto ser mais transmissíveis quanto apresentarem maior resistência ao sistema imunológico humano, vacinas, medicamentos, etc.

Felizmente, a comunidade científica, junto das tecnologias de informática e engenharia genética, disponibiliza o acesso público a dados genômicos de qualquer ser vivo através da plataforma GenBank, mantida pelo NCBI, antes mesmo do surgimento da pandemia. Utilizando-se dessas técnicas de bioinformática, foi possível sequenciar o genoma do novo coronavírus (WU et al., 2020), assim como suas cepas (BEDFORD et al., 2020). O monitoramento dessas mutações é importante para predição de novas mutações, saber se uma vacina desenvolvida será suficiente para imunizar, reforçar a dose ou desenvolver uma nova vacina (MARTINS, 2022).

No entanto, esses dados genômicos por si só não apresentam nenhuma informação útil. São sequências de letras que representam cada uma uma base nitrogenada do DNA ou RNA: G para Guanina, C para Citosina, A para Adenina e T para Timina (no caso do RNA a Timina é substituída por U de Uracila). São necessárias outras estratégias para extrair alguma informação relevante. Uma dessas é a decodificação de cada trinca de letras (chamada de códon) em um aminoácido que combinado com outros formam uma proteína. Outra estratégia, considerando os princípios de evolução das espécies, é comparar sequências genômicas distintas e obter uma árvore filogenética a partir disso. Essa última abordagem é interessante para o entendimento das mutações e predição de novas mutações de uma espécie analisada. Portanto, esse será o foco desse trabalho.

Uma árvore filogenética pressupõe a existência de um acestral comum e suas linhagens. Isso se desenvolve recursivamente formando uma árvore. Na biologia também é utilizado o termo Cladística. Em CUI, J. et al. (2007) se modelou uma árvore filogenética para a compreensão de mutações do coronavírus presente em morcegos.

Dado um conjunto de sequências genômicas, se comparadas em pares a fim de obter a similaridade ou a diferença entre elas, é possível abstrair um grafo. Nesse grafo, os nós são as sequências e as arestas são ponderadas pelas similaridades/diferenças entre as sequências. Também é plausível dizer que quanto menor a distância entre dois nós, maiores as chances de

estabelecerem uma relação de ancestral comum e linhagem. Minimizar a soma dos pesos das arestas de forma que todos os nós sejam visitados recai num problema de árvore de extensão mínima (MST) e consequentemente se obtém uma árvore. Com isso, uma árvore de extensão mínima é uma boa aproximação para uma árvore filogenética.

A proposta desse trabalho é modelar as cepas do SARS-CoV-2 em uma árvore de extensão mínima, originada de um grafo inicial, de modo que possa facilitar a compreensão e eventual predição de novas mutações em trabalhos futuros.

Inicialmente a MST será calculada pelos algoritmos determinísticos conhecidos na literatura, no caso Prim e Kruskal. Como a comparação de todas as sequências tem complexidade $\mathcal{O}(n^2(\overline{k})^2)$, sendo n o número de sequências e k o tamanho médio das sequências, o tempo de execução desses métodos tendem a ser proibitivos para grafos grandes. Portanto, uma abordagem heurística para uma MST aproximada será proposta. Dessa maneira a complexidade passa a ser $\mathcal{O}(n(\overline{f}(n) + \overline{r}(n))(\overline{k})^2)$, sendo f o número médio de folhas visitadas e r o número médio de nós visitados (com exceção das folhas) até encontra o ótimo local. O algoritmo heurístico proposto será avaliado quanto ao seu tempo de execução e comparado com as abordagens exatas.

As análises serão realizadas para (1) dados do Brasil e (2) dados de outros paises (i.e. Argentina, Chile, Colombia, Equador, Peru, Uruguai e Venezuela).

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Obter uma árvore filogenética das cepas dos vírus SARS-CoV-2, por meio da árvore de extensão mínima aplicada em um grafo. As arestas são obtidas por similaridade entre sequências genéticas, usando a distância de Levenshtein.

2 REVISÃO DE LITERATURA

2.1 BIOLOGIA DO CORONAVÍRUS

O SARS-CoV-2 é um betacoronavírus, um dos quatro tipos de coronavírus. Como todo vírus, possui um código genético. Esse código genético é um RNA de cadeia simples positiva. Os elementos constituintes desse código são as bases nitrogenadas: Uracila, Guanina, Citosina e Adenina (V'KOVSKI et al., 2020).

De maneira geral, em qualquer ácido nucléico, cada trinca de bases nitrogenadas forma um códon. Cada códon codifica apenas um aminoácido. No entanto, diferentes códons podem codificar o mesmo aminoácido. Uma cadeia de aminoácidos forma uma proteína.

Uma das proteínas codificadas pelo coronavírus é a proteína Spike, que pode ser de dois tipos, S1 e S2. Essa proteína situa-se no envelope do vírus e auxilia na endocitose, permitindo as ligações com os receptores ECA-2 e TMTRSS2. Esses receptores estão presentes nas células pneumócitas e macrófagas pulmonares. Felizmente, anticorpos podem desestabilizar a ligação entre a proteína S com ECA-2 (STELATO, 2022).

Uma vez dentro das células, o vírus se utiliza de proteínas N de sua carga viral e outras presente no seu capsídio para replicar-se e para a inibição do sistema imune inato do paciente (STELATO, 2022).

As tecnologias presentes na maioria das vacinas têm como princípio induzir uma resposta imunológica do corpo humano à proteína Spike (RIBEIRO et al., 2021). Por isso, caso haja uma nova variante do vírus com mutação nessa proteína, como por exemplo a ômicron, a eficácia da vacina pode ser dificultada ou até mesmo inibida, tornando a variante mais infecciosa, requerendo uma atualização da mesma (MCCALLUM et al., 2022). Ao mesmo tempo, é válido destacar os esforços para se criar uma vacina universal ou pan-variante que abranja essas possíveis mutações, mas para isso é necessário ter conhecimento de uma parte estável do código genético do vírus, ou seja, que raramente ou nunca sofre mutação (FAPESP, 2022).

Dessa forma, conclui-se que é importante o monitoramento dessas mutações. Não somente das partes do material genético que sofrem mutações (para verificar se haverá modificação na proteína Spike) como também das partes mais estáveis (que torna viável uma vacina universal).

2.2 DISTÂNCIA DE LEVENSHTEIN

Para saber o quão diferente são dois códigos genéticos distintos, é possível reduzir o problema para uma comparação entre duas strings de tamanhos distintos. A solução na literatura encontrada para isso é a distância de Levenshtein (NIST, 2019).

Essa distância leva em conta o número de operações necessárias para transformar uma

string em outra. Essas operações incluem: Deleção, substituição e inserção. Por exemplo, a distância entre a palavra 'metano' e 'etanal' é três.

- 'etano' Operação de deleção do m.
- 'etana' Substituição do o por a.
- 'etanal' Inserção do l.

O algoritmo inicia-se criando uma tabela com o número de linhas m equivalente ao número de caracteres da primeira palavra mais um e o número de colunas n equivalente ao número de caracteres da segunda palavra mais um. A primeira linha é preenchida de 0 até n, sendo o valor o respectivo índice da coluna. A primeira coluna é preenchida de 0 até m, sendo o valor o respectivo índice da linha. Depois, itera-se para cada linha e cada coluna, a partir da segunda posição de ambos(linha 1, coluna 1). Para cada posição (i,j), calcula-se o custo e verifica a vizinhança elegendo aquela que tiver o menor valor. O custo é calculado comparando o caractere da primeira palavra, cujo índice equivale a posição da coluna corrente, com o caractere da segunda palavra, cujo índice equivale a posição da linha corrente. Se forem iguais o custo é zero, caso contrário é um. Na vizinhança, está contida na posição anterior de coluna e linha a operação equivalente de substituição, na posição atual da linha e coluna anterior a operação equivalente de inserção, na posição da linha anterior e coluna atual a operação equivalente de deleção. Adiciona-se um para operações de deleção e inserção, enquanto que para operação de substituição adiciona-se o custo. Dentre essas operações, elege-se a que tiver menor valor e que ocupará a posição da linha e coluna atual. A última linha na última coluna estará o valor correto da distância de Levenshtein. Esses passos se encontram no pseudocódigo (Algoritmo 1).

Para elucidar melhor o funcionamento desse algoritmo, vamos a um exemplo, usando novamente as palavras 'metano' e 'etanal'. A primeira palavra é metano e dará o número de linhas mais um, totalizando 7 linhas. A segunda palavra é etanal e dará o número de colunas mais um, totalizando 7 linhas também.

		e	t	a	n	a	l
	0	1	2	3	4	5	6
m	1						
e	2						
t	3						
a	4						
n	5						
o	6						

Figura 1 – Levenshtein - Passo 01

A primeira linha é preenchida de 0 a 6. Isso é o equivalente ao número de operações de deleção para chegar ao caractere vazio, para cada conjunto de letras. Exemplo: a distância entre 'eta' e vazio é três, pois é necessário deletar três letras.

Algoritmo 1: Lenvenshtein-Distance (str1, str2)

```
m ← tamanho de str1
n ← tamanho de str2
Criar uma tabela de m+1 linhas e n+1 colunas
for cada i de 0 até m do
    tab[i, 0] ← i
end
for cada j de 0 até n do
    tab[0, j] ← j
end
for cada i de 1 até m do
    for cada j de 1 até n do
        custo ← 1 if str1[i] == str2[j] then
            custo ← 0
        end
        deletar ← tab[i − 1, j] + 1
        inserir ← tab[i, j − 1] + 1
        substituir ← tab[i − 1, j − 1] + custo
        tab[i,j] ← MIN(deletar, inserir, substituir)
    end
end
return tab[m,n]
```

A primeira coluna é preenchida de 0 a 6. Isso é o equivalente ao número de operações de inserção a partir do caractere vazio, para cada conjunto de letras. Exemplo: a distância entre vazio e 'metan' é cinco, ou seja, deve-se inserir cinco letras.

A iteração começa na linha 2 coluna 2. Para cada iteração verifica-se a vizinhança, acrescenta o custo para substituição, uma unidade para inserção e outra para deleção, elege o menor.

No exemplo, Na linha 2 e coluna 2 a letra da primeira palavra (m) é diferente da letra da segunda palavra (e), logo o custo é um. A substituição passa a ser 1(0 + 1), deleção 2(1 + 1) e inserção 2(1 + 1). O valor consolidado na tabela é 1. Esse valor é coerente com a distância de 'm' para 'e', que envolve uma operação de substituição apenas.

		e	t	a	n	a	l
	0	1	2	3	4	5	6
m	1	1					
e	2						
t	3						
a	4						
n	5						
o	6						

Figura 2 – Levenshtein - Passo 02a

No outro exemplo, na linha 3 e coluna 2, a letra da primeira palavra coincide com a letra da segunda palavra (e), logo o custo é zero. Substituição passa a ser 1 (1 + 0), deleção 2 (1 + 1) e inserção 3 (2 + 1). O valor consolidado é 1. Esse valor representa a distância de 'me' para 'e', bastando apenas uma operação.

		e	t	a	n	a	l
	0	1	2	3	4	5	6
m	1	1	2	3	4	5	6
e	2	1					
t	3						
a	4						
n	5						
o	6						

Figura 3 – Levenshtein - Passo 02b

Esse processo se repete até a última posição da tabela e o valor contido nela é retornado.

		e	t	a	n	a	l
	0	1	2	3	4	5	6
m	1	1	2	3	4	5	6
e	2	1	2	3	4	5	6
t	3	2	1	2	3	4	5
a	4	3	2	1	2	2	3
n	5	4	3	2	1	2	3
o	6	5	4	3	2	2	3

Figura 4 – Levenshtein - Passo 03

2.3 GRAFOS

Um grafo é uma estrutura de dados formada por nós (também chamados de vértices) e arestas. Um nó é um ponto no grafo. Uma aresta é uma ligação entre dois nós, representada por um segmento de reta. Se o nó inicial e final for o mesmo a aresta é um laço, se o nó inicial e final forem distintos é um arco. Arestas podem ter pesos, assim como os nós.

Um grafo pode ser direcionado, se e somente se for importante a ordem do par de nós que formam uma aresta. Se as arestas em um grafo podem ser representadas independente da ordem do par de nós, é dito que o grafo é não direcionado.

Dada uma sequência de nós pertencentes a um grafo, se para qualquer nó v pertencente a essa sequência e existindo um nó consequente w, existe um arco v-w pertencente ao grafo, é dito que essa sequência é um passeio. Se nenhum arco se repete na sequência esse passeio é um caminho (FEOFILOFF, 2017).

Dado um caminho pertencente a um grafo, se o nó inicial corrresponder com o nó final, tem-se um ciclo (FEOFILOFF, 2017).

Dado um conjunto de nós V, se para qualquer par de nós pertencentes a esse conjunto existe pelo menos um caminho que os conectem, é dito que o grafo é conexo. Caso contrário o grafo é desconexo.

As características desejadas para esse trabalho, é que o grafo seja não direcionado e conexo. Também é desejado que a soma dos pesos das arestas seja o menor possível. Com isso, o problema recai em uma árvore de extensão mínima. Depois de aplicado o algoritmo MST, espera-se que todos os ciclos possíveis no grafo cessem de existir.

2.4 ÁRVORE DE EXTENSÃO MÍNIMA

Primeiramente, é necessário um grafo conexo, não direcionado e de arestas ponderadas. Denota-se por V seu conjunto de vértices. Uma árvore de extensão mínima (MST) é um subgrafo que conecta todos os vértices pertencentes a V e ao mesmo tempo minimiza a soma dos pesos das arestas (QUITZAU, 2003). Consequentemente, esse grafo é conexo, não possui ciclos e com isso pode-se dizer que sua estrutura de dados é uma árvore (BARBOSA, 1975). Também há um caminho único para qualquer par de vértices nessa árvore (SZWARCFITER, 1988). Além disso, com a minimização dos pesos das arestas, é o menor caminho possível no grafo original. Veja a versão generalizada de um algoritmo MST em pseudocódigo (Algoritmo 2).

Algoritmo 2: GENERIC-MST(G,w)

```
A ← {}
while A não forma uma MST do
    encontre uma aresta (u,v) que é segura para A
    A ← A união (u,v)
end
return A
```

Na literatura já existem algoritmos que calculam a árvore de extensão mínima: Algoritmo de Prim e Algoritmo de Kruskal.

A ideia do algoritmo de Prim (Algoritmo 3) é partir de uma árvore que cresce a cada iteração. Cada nó é iniciado com uma chave de valor infinito e um pai com valor nulo. A raiz no entanto inicia-se com valor de chave zero. Os vértices também são armazenados numa fila de prioridade denotada por Q. Depois inicia-se um laço que só para quando essa fila estiver vazia. A cada iteração é extraído um nó, denotado por u, que possui a menor chave em Q. Para cada nó adjacente v de u, verifica se o mesmo pertence a Q e se a aresta $w(u,v)$ é menor do que a chave do nó adjacente. Em caso afirmativo, o pai de v será u e a nova chave de v será a aresta $w(u,v)$. No final retorna-se o conjunto de nós e os respectivos pais (QUITZAU, 2003).

Algoritmo 3: MST-PRIM(G,w,r)

```
for cada u ∈ V[G] do
    chave[u] ← infinito
    pai[u] ← NULL
end
chave[r] ← 0
Q ← V[G]
while Q != {} do
    u ← EXTRACT − MIN(Q)
    for cada v em Adj[u] do
        if v ∈ Q e w(u,v) < chave[v] then
            pai[v] ← u
            chave[v] ← w(u,v)
        end
    end
end
```

O algoritmo de Kruskal (Algoritmo 4) conecta árvores de uma floresta, que inicialmente são formadas por um vértice cada. Primeiro, ordenam-se as arestas em ordem crescente de peso. Em um laço, as arestas são extraídas em ordem crescente e a cada iteração, se a aresta conecta árvores distintas, então ela é adicionada a árvore final A e as árvores são atualizadas na função UNION. No final a árvore otimizada A é retornada.

Algoritmo 4: MST-KRUSKAL(G,w)

```
A ← {}
for cada vértice v em V[G] do
    MAKE-SET(v)
end
Ordene as arestas de E em ordem crescente de peso (w)
for cada aresta (u,v), tomadas em ordem crescente de peso (w) do
    if FIND-SET(u) != FIND-SET(v) then
        A ← A ∪ (u,v)
        UNION(u,v)
    end
end
return A
```

2.5 TRABALHOS CORRELATOS

Em (SPADA et al., 2004) utilizou-se uma árvore de extensão mínima para identificar o histórico de transmissão do vírus da Hepatite C num surto envolvendo 5 crianças atendidas em um ambulatório de oncologia-hematologia pediátrica entre 1992 e 2000. A partir de amostras de sangue coletadas, realizou-se a análise de sequência e análise de árvore filogenética da região hipervariável 1 do gene E2. O MST, aplicado aos dados moleculares, juntamente com os

dados clínico-epidemiológicos, permitiu identificar a origem do surto e a cadeia de transmissão paciente-paciente mais provável.

Em (SALIPANTE, 2011) alega um problema de se usar MST. Argumenta que MST selecionado arbitrariamente representa uma das muitas soluções ótimas e que por isso faz-se necessário o uso de métricas estatísticas para avaliar a credibilidade das estimativas de MST. Essas soluções ótimas surgem quando há um empate nos pesos das arestas a serem escolhidas. Para selecionar o MST mais confiável utilizou-se uma métrica de bootstrapping.

Em (EBERHARD, 2022) compara DNA mitocondrial de diferentes espécies para criar uma árvore filogenética. Há uma crítica à distância de Levenshtein, alegando que a mesma é muito lenta para grandes sequências genéticas (superior a 40 mil bases). Por isso, utilizou-se agrupamento hierárquico de ligação média. No entanto, essa estratégia funciona apenas para espécies que possuem grandes distâncias na árvore, ou seja, em nível macro.

Em (ABDOLLAHI et al., 2022) busca reconstruir a história evolutiva das linhagens de células B. A proposta é um algoritmo, de nome ClonalTree, que incorpora abundâncias de genótipos em algoritmos MSTs. Consequentemente há um tempo de execução menor. O algoritmo MST utilizado é o Prim modificado para o problema. A dissimilaridade entre as sequências é calculada utilizando a distância Hamming normalizada, pois alega-se ser menos custosa em termos de memória e tempo para o problema, se comparada a distância de Levenshtein.

Como na literatura são apresentados problemas de performance e memória ao se usar a distância de Levenshtein, optou-se por utilizar uma versão melhorada do algoritmo em Python, da biblioteca Polyleven (SEIJI, 2021).

Em relação as MSTs, o foco não é a correta linhagem dos vírus e sim quais mutações tem mais chances de ocorrer. Dessa forma, não é preciso considerar as outra soluções ótimas globais para o problema.

3 MATERIAS E MÉTODOS

3.1 MATERIAIS

Foram utilizados dados públicos da plataforma online GenBank mantida pela NCBI. A mesma plataforma disponibiliza sequenciamentos genômicos em arquivos do tipo texto e de extensão FASTA. Considerando que os dados podem estar atualizados a partir da data do experimento, para que os resultados sejam similares aos apresentados, recomenda-se utilizar o arquivo mantido no repositório github (BARRETA, 2022a).

Esses dados contém informações relativas ao nome de acesso, informações do ser vivo analisado, data da publicação, país e a sequência genômica. Um fragmento desses dados pode ser visto abaixo:

```
>MZ169910.1 |Severe acute respiratory syndrome coronavirus 2
isolate SARS-CoV-2/human/BRA/1061/2021, complete genome|
2021-05-12T00:00:00Z|Brazil|1061
ATTAAAGGTTTATACCTTCCCAGGTAACAAACCAACCAACTTTCGATCTCTTGTAGATCT
GTTCTCTAAACGAACTTTAAAATCTGTGTGGCTGTCACTCGGCTGCATGCTTAGTGCACT.....
```

As sequências genômicas do SARS-CoV-2 apresentam uma média de aproximadamente 12 mil códons (BARRETA, 2022d) ou 36 mil bases nitrogenadas. Essas estão agrupadas por país: Brasil (NCBI, 2022a), Argentina (NCBI, 2022b), Chile (NCBI, 2022c), Colômbia (NCBI, 2022c), Equador (NCBI, 2022d), Peru (NCBI, 2022e), Uruguai (NCBI, 2022f) e Venezuela (NCBI, 2022g). O número de sequências por país pode ser visto na Tabela 1.

Tabela 1 – Número de sequências por país

país	Número de sequências
Colômbia	2
Equador	4
Venezuela	24
Uruguai	32
Argentina	34
Brasil	132
Peru	142
Chile	378

Fonte: (BARRETA, 2022b)

3.2 MÉTODOS

Os dados foram carregados e tratados no Google Colab, utlizando a linguagem de programação Python e os dados genômicos do Brasil (NCBI, 2022a). As sequências de DNA de cada mutação do vírus SARS-CoV-2 foram transcritas para aminoácidos. Com isso, reduziu-se o número de caracteres para representá-los.

Tendo os dados devidamente tratados, o próximo passo foi calcular a distância de Levenshtein entre duas sequências de aminoácidos de vírus distintos. A comparação foi realizada de todos com todos e calculou-se o tempo de execução necessário.

É razoavelmente plausível abstrair um grafo, se for considerado que a distância de Levenshtein é uma aresta ponderada e cada mutação um nó. Dessa forma, obtém-se um grafo completo, em que todo nó se conecta a qualquer nó. No entanto ainda é preciso obter uma árvore de extensão mínima a partir desse grafo. Com isso, é possível aproximar-se da árvore filogenética (NCBI, 2022h). Esse procedimento pode ser observado na Figura 5.

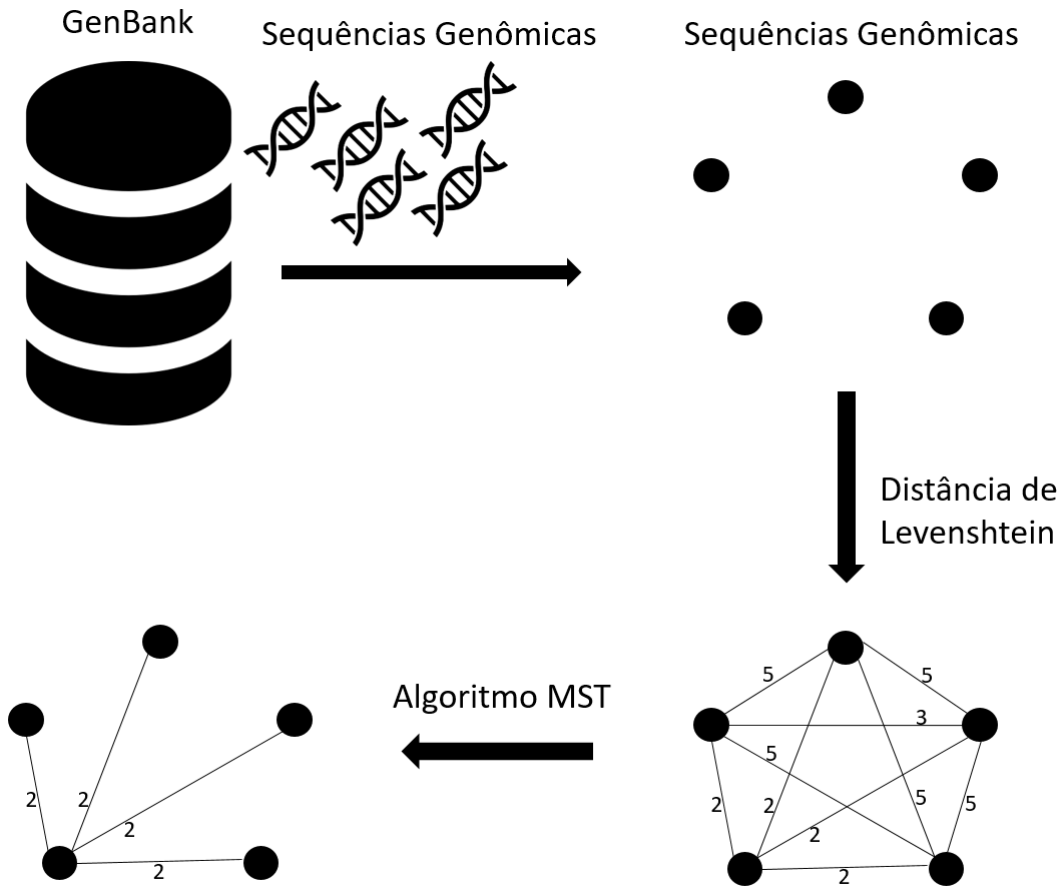

Figura 5 – Fluxograma do algoritmo determinístico

Para obter a árvore de extensão mínima foram utilizados dois algoritmos: Prim e Kruskal. Como a soma das arestas apresentadas por cada algoritmo foi a mesma, é razoável

deduzir que foram implementados corretamente. Também verificou-se o tempo de execução levado por cada algoritmo.

Após a obtenção da MST, uma análise visual da árvore obtida é realizada.

Como os algoritmos exatos são proibitivos, em termos de tempo computacional, para grafos grandes, um método heurístico foi proposto. Para validar esse algoritmo novo, utilizou-se conjuntos de dados por país. As métricas do tempo de execução, número de sequências e soma mínima das arestas são utilizadas na análise final.

4 RESULTADOS E DISCUSSÕES

4.1 MST VIA MÉTODOS EXATOS

No primeiro experimento considerou-se somente os dados genômicos do Brasil listados na Tabela 6.

Cada índice equivale ao nó correspondente na árvore plotada no final. O nome é o código para consulta no GenBank.

A métrica principal considerada foi o tempo de execução em determinadas funções do algoritmo que foram consideradas críticas. A primeira função calcula a distância de Levenshtein (Algoritmo 1) entre todos os pares de sequencia. A segunda é o algoritmo de Prim (Algoritmo 3), usado para calcular a árvore de extensão mínima (MST). A terceira é o algoritmo de Kruskal (Algoritmo 4), também utilizado para calcular MST. Os resultados são apresentados na Tabela 2. Os códigos desse experimento e demais encontram-se disponíveis em (BARRETA, 2022d).

Tabela 2 – Tempo de execução em cada função

Distância de Lenvenshtein	257 segundos
Algoritmo de Prim	1,8 segundos
Algoritmo de Kruskal	0,04 segundos

Fonte: (BARRETA, 2022b)

Ambos os algoritmos destinados ao cálculo da MST apresentaram uma solução ótima global. No entanto, o algoritmo de Kruskal demonstrou melhor desempenho de tempo para esse problema, restando descobrir se o desempenho se manterá caso o problema escale para um número maior de sequências genéticas.

O trecho de código mais custoso é o da função que calcula as distâncias entre as sequências genéticas. Mesmo implementando um algoritmo de Levenshtein otimizado, a estratégia de comparar par a par pode ser um impeditivo caso seja necessário escalar o problema. Se considerarmos uma combinação simples par a par, sendo n o número de sequências genômicas e sabendo que o tempo necessário para executar o algoritmo é 257 segundos para 132 sequências, temos:

$$\text{Combinação simples : } \binom{n}{2} = (n)(n-1)/2 = (n^2 - n)/2 \tag{1}$$

$$\text{Combinações possíveis : } \binom{132}{2} = (132^2 - 132)/2 = 8646 \tag{2}$$

$$\text{tempo de execução por sequência genômica : } \frac{\text{Tempo de execução}}{\text{Combinações possíveis}} = \frac{257}{8646} \approx 0{,}03s \tag{3}$$

Sabendo que a distância de Levenshtein envolve a varredura de uma matriz, é possível aproximar a complexidade para $\mathcal{O}((\overline{k})^2)$, sendo k o tamanho médio das sequências genéticas.

Além disso, a combinação simples par a par resulta numa complexidade $\mathcal{O}(n^2(\overline{k})^2)$, sendo n o número de sequências. Com isso é possível percerber que há um gargalo no tempo de execução, impedindo a escalabilidade para um número maior de sequências.

Há duas formas de melhorar o tempo de execução nessa função:

- Considerar um cálculo de distância alternativo.
- Evitar ter que comparar todas as combinações possíveis.

Essa segunda opção foi a considerada nesse trabalho, já que o cálculo da distância já é otimizado (SEIJI, 2018).

4.2 MST VIA MÉTODO HEURÍSTICO

Para contornar o problema de tempo computacional do cálculo da distância de Levenshtein, foi realizado um segundo experimento utilizando um algoritmo heurístico. O código desse algoritmo está disponível em (BARRETA, 2022c).

Conforme o fluxograma da Figura 6 a idéia é obter a MST sem a necessidade de calcular todas as distâncias par a par de sequencias genéticas. Partindo de uma árvore com apenas dois nós e uma aresta os unindo, compara-se um novo nó para cada um dos n-2 nós restantes, sendo n o número total de nós do grafo original. Para cada novo nó, primeiro verifica-se qual a distância do mesmo para os outros nós de grau um, escolhendo aquele com a menor distância e incluindo os outros na lista de nós visitados. Escolhido o nó, percorre-se o ramo até encontrar um ótimo local satisfatório. Para cada iteração aplica-se Kruskal no subgrafo em construção e atualiza o grau dos nós. O processo se repete para o restante dos nós. Esses passos estão descritos no pseudocódigo apresentado no Algoritmo 5.

Algoritmo 5: Algoritmo-Heurístico

```
tam ← quantidade de sequencias
V ← [0,1]
graus ← [1,1]
distancia ← OBTER-DISTANCIA(0,1)
E ← [[distancia, 0, 1]]
for cada i de 2 até tam do
    visitados ← CANDIDATOS-GRAU-UM(V, grau)
    escolhido, distancia ← ESCOLHER(visitados,i)
    V,E ← PERCORRER-RAMO(escolhido,i,E,distancia, visitados)
    E = MST-KRUSKAL(V,E)
    ATUALIZAR-GRAUS(V,E)
end
```

Para percorrer os nós do ramo escolhido, inicia-se outro laço, em que cada iteração obtém os nós adjacentes não visitados e escolhe o nó com a distância menor ou igual em relação a todos os nós visitados. O critério de parada desse laço é quando todos os nós adjacentes não visitados possuírem distância maior do que a menor distância encontrada ou quando não haver

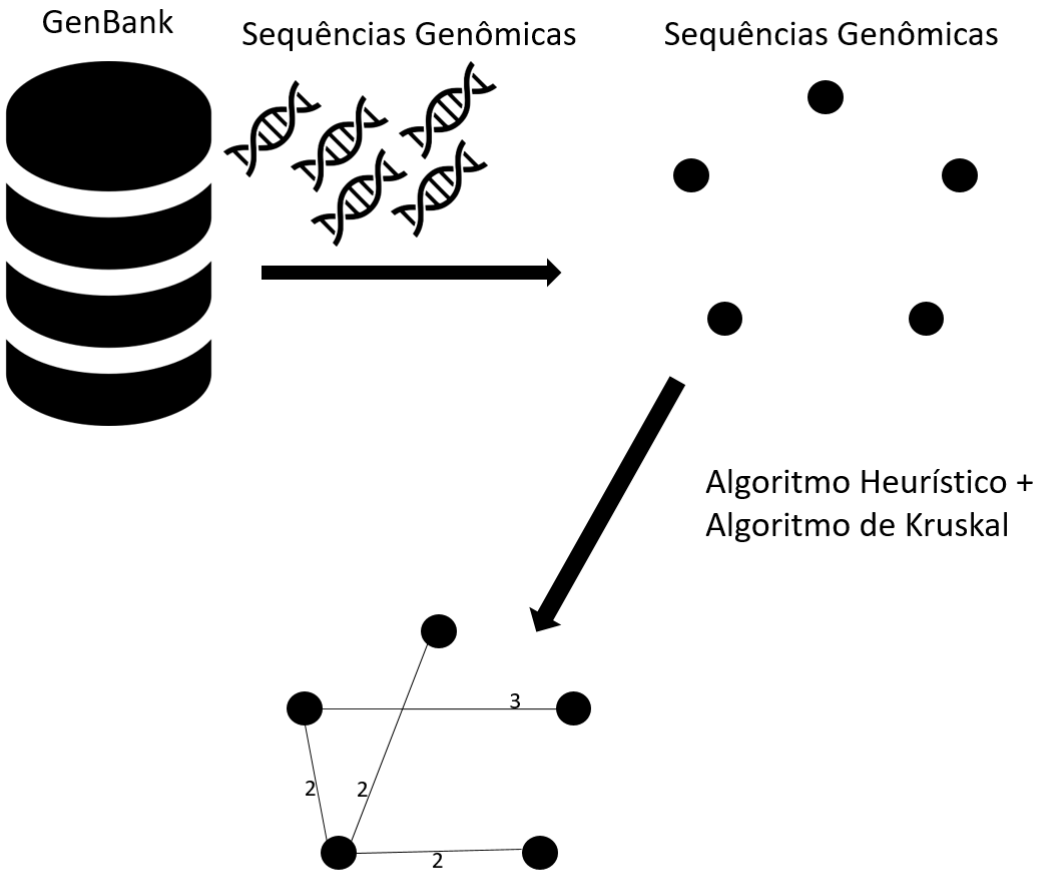

Figura 6 – Fluxograma do algoritmo heurístico

mais nós adjacentes não visitados. Encerrado o laço, atualiza-se a árvore com as distâncias de todos os nós visitados em relação ao nó avaliado e adiciona-se o nó avaliado no conjunto de nós do subgrafo. Por último retorna-se o conjunto de vértices e arestas. Esses passos estão descritos no pseudocódigo (Algoritmo 6).

A complexidade teórica desse algoritmo pode ser aproximada por $\mathcal{O}(n(\overline{f}(n)+\overline{r}(n))\overline{k}^2)$, sendo k o tamanho médio da sequência e n o número de sequências. Para cada iteração do algoritmo, de 1 até n, há um número de folhas visitadas, representado por f, e um número de nós visitados (com exceção das folhas) até o ótimo local, representado por r. Para obter $n(\overline{f}(n)+\overline{r}(n))$, segue que:

$$\sum_{i=1}^{n} f(i)+r(i) = \sum_{i=1}^{n} f(i) + \sum_{i=1}^{n} r(i) = n\overline{f}(n) + n\overline{r}(n) = n(\overline{f}(n)+\overline{r}(n))$$

Também é válido que $\overline{f}(n)+\overline{r}(n) \leq n$, logo a complexidade pode chegar a $\mathcal{O}(n^2(\overline{k})^2)$ na pior das hipóteses.

Para efeitos de comparação de resultados as árvores MST obtidas, com e sem utilização do algoritmo heurístico, são apresentados nas Figuras 7 e 8 respectivamente. É possível observar

Algoritmo 6: PERCORRER-RAMO(escolhido,i,E,distancia, visitados)

```
while True do
    adjacentes ← OBTER-ADJACENTES(escolhido,E)
    candidatos ← adjacentes não visitados
    if não tem adjacentes não visitados then
        sair do loop
    end
    visitados ← visitados ∪ candidatos
    novo_escolhido, nova_distancia ← ESCOLHER(candidatos,i)
    if nova_distancia > distancia then
        sair do loop
    end
    escolhido, distancia ← novo_escolhido, nova_distancia
end
V,E ← ATUALIZAR-ARVORE(V,E,escolhido,i,visitados)
return V,E
```

uma semelhança entre as árvores.

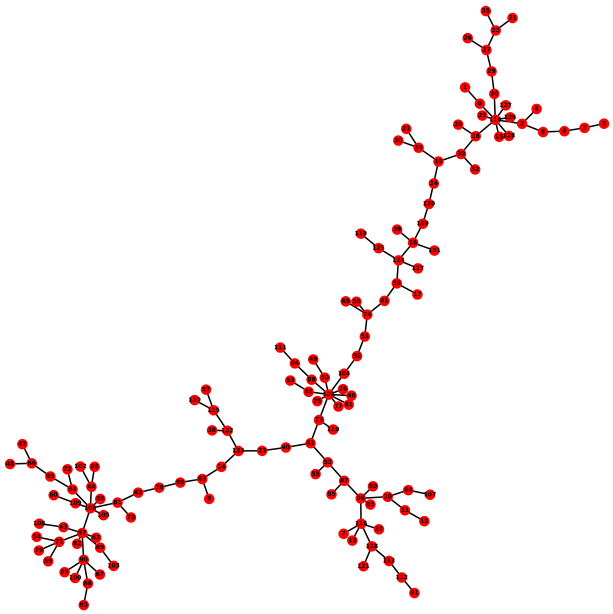

Figura 7 – Algoritmo Kruskal aplicado nos dados genômicos do Brasil (BARRETA, 2022d)

Para avaliar a viabilidade de custo computacional do algoritmo heurístico proposto, utilizou-se as métricas do número de sequências por país, soma mínima das arestas, ambos apresentados na Tabela 3 e tempo de execução apresentado na Tabela 4.

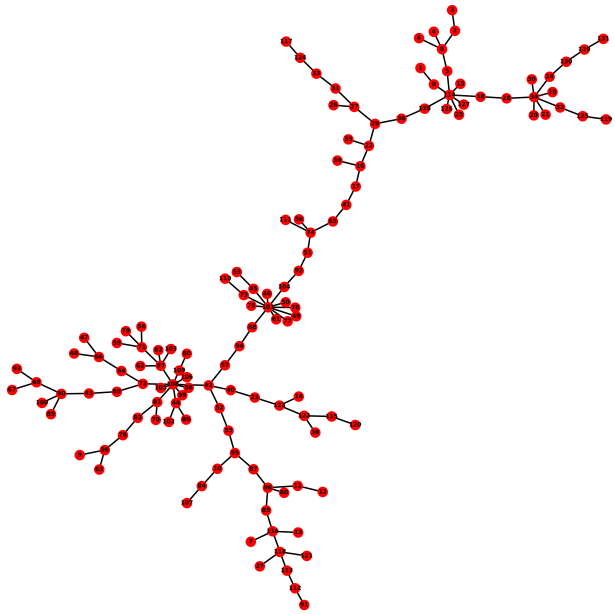

Figura 8 – Algoritmo heurístico combinado com Kruskal aplicado nos dados genômicos do Brasil (BARRETA, 2022c)

Tabela 3 – Soma mínima de arestas das amostras

país	Número de sequências	Soma mínima de arestas (apenas Kruskal)	Soma mínima de arestas (com algoritmo heurístico)	Erro (%)
Colômbia	2	64.0	64.0	0.0
Equador	4	53.0	53.0	0.0
Venezuela	24	19517.0	19517.0	
Uruguai	32	21380.0	30905.0	44.6
Argentina	34	20010.0	20082.0	0.4
Brasil	132	22400.0	22571.0	0.8
Peru	142	25002.0	25724.0	2.9
Chile	378	26472.0	27627.0	4.4

Fonte: (BARRETA, 2022b)

O erro da soma mínima foi calculado da seguinte forma:

K : Soma mínima (apenas Kruskal) (4)

H : Soma mínima (algoritmo heurístico) (5)

Erro da soma mínima : $\frac{H - K}{K}$ (6)

(7)

Com isso, o erro médio foi de 6%. Isso significa que a taxa de acerto é, na média, 94%.

Tabela 4 – Tempo de execução das amostras

país	Número de sequências	tempo de execução (seg) (apenas Kruskal)	Tempo de execução (seg) (com algoritmo heurístico)
Colômbia	2	0.023	0.024
Equador	4	0.137	0.182
Venezuela	24	7.774	5.948
Uruguai	32	13.394	5.31
Argentina	34	18.314	13.649
Brasil	132	252.376	129.251
Peru	142	241.976	129.398
Chile	378	1926.472	857.321

Fonte: (BARRETA, 2022b)

É uma boa aproximação se comparado aos métodos de aprendizagem de máquina tradicionais.

O cálculo relativo do tempo de execução (Tabela 5) foi calculado da seguinte forma:

S : Tempo de execução (apenas Kruskal) (8)

F : Tempo de execução (algoritmo heurístico) (9)

Tempo de execução relativo : $\frac{F}{S}$ (10)

(11)

O tempo de execução com algoritmo heurístico apresentou em média 72% do tempo de execução do código contendo apenas o algoritmo Kruskal. No entanto, é válido destacar que há uma tendência para que o tempo de execução relativa diminua, a medida que mais nós são acrescentados, conforme pode ser observado na Tabela 5 e também na Figura 9.

Tabela 5 – Tendência do Tempo de Execução Relativo

país	Número de sequências	Tempo de execução Relativo
Colômbia	2	1.03
Equador	4	1.33
Venezuela	24	0.77
Uruguai	32	0.4
Argentina	34	0.75
Brasil	132	0.51
Peru	142	0.53
Chile	378	0.45

Fonte: (BARRETA, 2022b)

No eixo horizontal da Figura 9 está o número de nós, enquanto que no eixo vertical superior está o tempo de execução e no eixo vertical inferior a soma de arestas. Os pontos em

azul indica o algoritmo exato em que foi usado apenas o algoritmo MST Kruskal. Os pontos em amarelo indica o algoritmo heurístico.

No eixo vertical inferior, da soma de arestas, é possível notar que há uma tendência para que os pontos se correlacionem, ao mesmo tempo em que a distância não varia muito, o que mantém o erro médio no patamar de 6%. No eixo vertical superior, do tempo de execução, por sua vez, tende a aumentar a distância a medida em que se avança positivamente ao longo do eixo horizontal. Com isso é razoável dizer que o tempo de execução relativo tende a ser cada vez menor para o algoritmo heurístico.

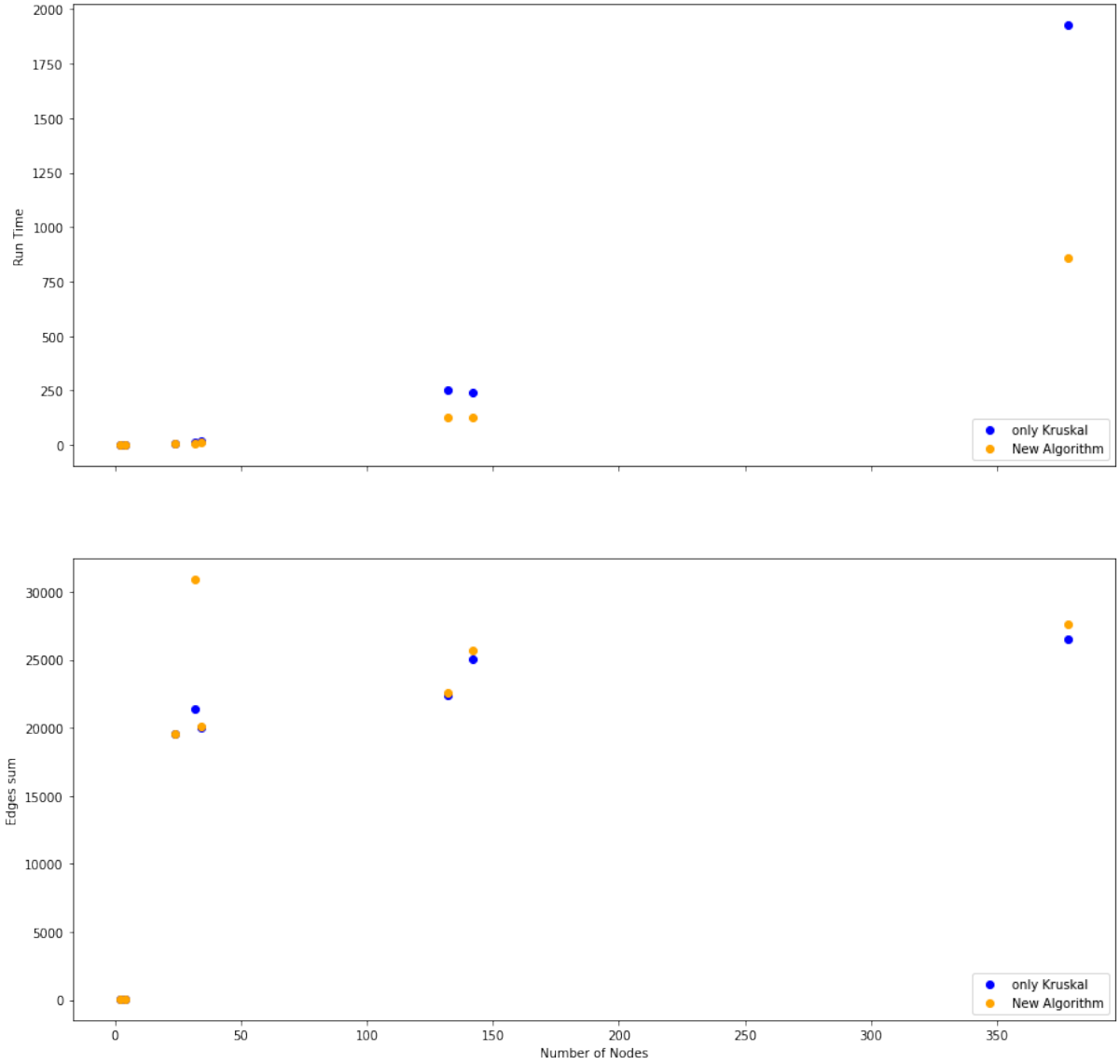

Figura 9 – Gráfico do tempo de execução e soma mínima das arestas (BARRETA, 2022b)

5 CONCLUSÃO

O objetivo desse trabalho foi modelar as cepas do vírus SARS-CoV-2 em uma árvore filogenética por meio da árvore de extensão mínima aplicada em um grafo.

Ao longo do desenvolvimento do experimento, utlizando uma combinação par a par de todas as sequências, verificou-se a impossibilidade de escalar o problema para um número maior de sequencias, pois o tempo de execução aumenta seguindo uma função polinomial grau dois. Por isso, foi proposto um algoritmo heurístico, que têm como premissas, reduzir o tempo de execução, assim como aproximar a solução ótima local da global.

Os resultados demostraram uma boa aproximação do método heurístico proposto se comparado com os métodos exatos. O tempo computacional da abordagem heurística se mostrou adequada.

No entanto, resta saber como o algoritmo irá perfomar se o problema escalar para um número maior de sequências. Baseando-se na informação interpretada dos dados, especula-se que o tempo de execução relativo tenda a reduzir para um número maior de sequências.

5.1 TRABALHOS FUTUROS

Os pontos que não foram abordados nesse trabalho são: escalar o problema para um número maior de sequências, cálculo de distância entre strings que seja uma alternativa para a distância de Levenshtein, predição de novas mutações e identificar os padrões das cepas vírus que permanecem estáveis.

Não houve uma revisão sistemática, para identificar outras alternativas para o cálculo de distância entre strings presentes na literatura.

Um termo comumente usado na literatura para o cálculo de distância é "Alinhamento de sequências". Com isso, é possível encontrar diversos algoritmos que substituem o algoritmo de Levenshtein: BLAST, Smith-Waterman, Needleman-Wunsch, Damerau–Levenshtein, distância de Hamming, etc.

Acredita-se que um algoritmo de aprendizado de máquina possa identificar padrões que se estabilizam nos trechos das sequências genômicas. Da mesma forma, predizer futuras mutações.

5.2 CONSIDERAÇÕES FINAIS

Os dados utilizados são de domínio público e estão disponibilizados para livre acesso. Com isso, seguindo a metodologia abordada, espera-se que ao tentar reproduzí-lo, obtenha os mesmos resultados.

Referências

ABDOLLAHI, N. et al. **Reconstructing the evolutionary history of a BCR lineage with minimum spanning tree and clonotype abundances**. 2022. <https://www.biorxiv.org/content/10.1101/2022.02.27.481992v1.full.pdf>. (Accessed on 03/21/2022). Citado na página 17.

BARBOSA, R. M. **Combinatória e Grafos**. 2. ed. São Paulo: Livraria Nobel: [s.n.], 1975. 196 p. Citado na página 15.

BARRETA, G. A. **GuilhermeBarreta/UTFPR_TCCE**. 2022. <https://github.com/GuilhermeBarreta/UTFPR_TCCE>. (Accessed on 03/30/2022). Citado na página 18.

BARRETA, G. A. **UTFPR_TCCE/codes/Statistics at main · GuilhermeBarreta/UTFPR_TCCE**. 2022. <https://github.com/GuilhermeBarreta/UTFPR_TCCE/tree/main/codes/Statistics>. (Accessed on 03/30/2022). Citado 6 vezes nas páginas 5, 18, 21, 25, 26 e 27.

BARRETA, G. A. **UTFPR_TCCE/TCC_Sequences_Brazil_Polyleven_Improved_Algorithm.ipynb at main · GuilhermeBarreta/UTFPR_TCCE**. 2022. <https://github.com/GuilhermeBarreta/UTFPR_TCCE/blob/main/codes/Improved_Algorithm/TCC_Sequences_Brazil_Polyleven_Improved_Algorithm.ipynb>. (Accessed on 03/30/2022). Citado 3 vezes nas páginas 5, 22 e 25.

BARRETA, G. A. **UTFPR_TCCE/TCC_Sequences_Brazil_Polyleven_Prim_Kruskal.ipynb at main · GuilhermeBarreta/UTFPR_TCCE**. 2022. <https://github.com/GuilhermeBarreta/UTFPR_TCCE/blob/main/codes/Polyleven_Kruskal/TCC_Sequences_Brazil_Polyleven_Prim_Kruskal.ipynb>. (Accessed on 03/30/2022). Citado 4 vezes nas páginas 5, 18, 21 e 24.

BARRETO, C. **Coronavírus: tudo o que você precisa saber sobre a nova pandemia - PEBMED**. 2020. <https://pebmed.com.br/coronavirus-tudo-o-que-voce-precisa-saber-sobre-a-nova-pandemia/>. (Accessed on 03/14/2022). Citado na página 9.

BEDFORD, T. et al. **Nextstrain / narratives / ncov / sit-rep / pt / 2020-01-30**. 2020. <https://nextstrain.org/narratives/ncov/sit-rep/pt/2020-01-30>. (Accessed on 12/12/2021). Citado na página 9.

CUI, J. et al. **Evolutionary Relationships between Bat Coronaviruses and Their Hosts - Volume 13, Number 10—October 2007 - Emerging Infectious Diseases journal - CDC**. 2007. <https://wwwnc.cdc.gov/eid/article/13/10/07-0448_article>. (Accessed on 12/12/2021). Citado na página 9.

EBERHARD, O. **Growing Phylogenetic Trees Using Hierarchical Clustering | bioRxiv**. 2022. <https://www.biorxiv.org/content/10.1101/2022.02.08.479565v1.abstract>. (Accessed on 03/21/2022). Citado na página 17.

FAPESP. **Mais um passo rumo à vacina universal contra os coronavírus : Revista Pesquisa Fapesp**. 2022. <https://revistapesquisa.fapesp.br/

mais-um-passo-rumo-a-vacina-universal/>. (Accessed on 03/17/2022). Citado na página 11.

FEOFILOFF, P. **Caminhos e ciclos em grafos**. 2017. <https://www.ime.usp.br/~pf/algoritmos_para_grafos/aulas/paths-and-cycles.html>. (Accessed on 03/18/2022). Citado 2 vezes nas páginas 14 e 15.

MARTINS, C. **Por que especialistas estimam vacinas anuais contra a covid - BBC News Brasil**. 2022. <https://www.bbc.com/portuguese/geral-60342928>. (Accessed on 03/09/2022). Citado na página 9.

MCCALLUM, M. et al. **Structural basis of SARS-CoV-2 Omicron immune evasion and receptor engagement**. 2022. <https://www.science.org/doi/10.1126/science.abn8652?utm_campaign=SciMag&utm_source=Social&utm_medium=Twitter>. (Accessed on 03/17/2022). Citado na página 11.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=SARS-CoV-2,%20taxid:2697049&Completeness_s=complete&Country_s=Brazil>. (Accessed on 01/30/2022). Citado 3 vezes nas páginas 18, 19 e 33.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=SARS-CoV-2,%20taxid:2697049&Completeness_s=complete&Country_s=Argentina>. (Accessed on 02/15/2022). Citado na página 18.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=SARS-CoV-2,%20taxid:2697049&Completeness_s=complete&Country_s=Chile>. (Accessed on 02/15/2022). Citado na página 18.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=SARS-CoV-2,%20taxid:2697049&Completeness_s=complete&Country_s=Ecuador>. (Accessed on 02/15/2022). Citado na página 18.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=SARS-CoV-2,%20taxid:2697049&Completeness_s=complete&Country_s=Peru>. (Accessed on 02/15/2022). Citado na página 18.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=SARS-CoV-2,%20taxid:2697049&Completeness_s=complete&Country_s=Uruguay>. (Accessed on 02/15/2022). Citado na página 18.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=SARS-CoV-2,%20taxid:2697049&Completeness_s=complete&Country_s=Venezuela>. (Accessed on 02/15/2022). Citado na página 18.

NCBI. **NCBI Virus**. 2022. <https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/tree?job_id=JSID_01_520781_130.14.18.128_9000__flu_blast>. (Accessed on 01/30/2022). Citado na página 19.

NIST. **Levenshtein distance**. 2019. <https://xlinux.nist.gov/dads/HTML/Levenshtein.html>. (Accessed on 12/12/2021). Citado na página 11.

QUITZAU, J. A. A. **Árvores Espalhadas Mínimas**. 2003. <https://www.ic.unicamp.br/~meidanis/courses/mo417/2003s1/aulas/2003-05-16.html>. (Accessed on 03/18/2022). Citado na página 15.

RIBEIRO, F. E. d. M. et al. **Como funcionam as vacinas para prevenção contra a COVID-19? - TelessaúdeRS-UFRGS**. 2021. <https://www.ufrgs.br/telessauders/posts_coronavirus/como-funcionam-as-vacinas-para-prevencao-contra-covid-19/#:~:text=RNA%20mensageiro%3A%20baseia%2Dse%20no,%2C%20DNA%2C%20v%C3%ADrus%20vivo%20atenuado.> (Accessed on 03/17/2022). Citado na página 11.

SALIPANTE, S. J. **Inadequacies of Minimum Spanning Trees in Molecular Epidemiology | Journal of Clinical Microbiology**. 2011. <https://journals.asm.org/doi/full/10.1128/JCM.00919-11>. (Accessed on 03/21/2022). Citado na página 17.

SEIJI, F. **Polyleven – Fast Pythonic Levenshtein Library — Polyleven**. 2018. <https://ceptord.net/20181215-polyleven.html>. (Accessed on 05/28/2022). Citado na página 22.

SEIJI, F. **polyleven · PyPI**. 2021. <https://pypi.org/project/polyleven/>. (Accessed on 03/27/2022). Citado na página 17.

SPADA, E. et al. **Use of the Minimum Spanning Tree Model for Molecular Epidemiological Investigation of a Nosocomial Outbreak of Hepatitis C Virus Infection | Journal of Clinical Microbiology**. 2004. <https://journals.asm.org/doi/full/10.1128/JCM.42.9.4230-4236.2004>. (Accessed on 03/21/2022). Citado na página 16.

STELATO, M. M. **Biologia e replicação do coronavírus: implicações para SARS-CoV-2 - Hospital PUC-Campinas**. 2022. <https://www.hospitalpuc-campinas.com.br/covid-19/biologia-e-replicacao-do-coronavirus-implicacoes-para-sars-cov-2/>. (Accessed on 03/17/2022). Citado na página 11.

SZWARCFITER, J. L. **Grafos e algoritmos computacionais**. Rio de Janeiro: Campus: [s.n.], 1988. 43-45 p. ISBN 85-7001-341-8. Citado na página 15.

V'KOVSKI, P. et al. **Coronavirus biology and replication: implications for SARS-CoV-2**. 2020. <https://www.hospitalpuc-campinas.com.br/wp-content/uploads/2020/12/Aritgo-microbio.pdf>. (Accessed on 12/12/2021). Citado na página 11.

WU, F. et al. **Severe acute respiratory syndrome coronavirus 2 isolate Wuhan-Hu-1, co - Nucleotide - NCBI**. 2020. <https://www.ncbi.nlm.nih.gov/nuccore/MN908947>. (Accessed on 12/12/2021). Citado na página 9.

ZHOU, P. et al. **A pneumonia outbreak associated with a new coronavirus of probable bat origin | Nature**. 2020. <https://www.nature.com/articles/s41586-020-2012-7>. (Accessed on 12/12/2021). Citado na página 9.

Apêndices

Tabela 6 – Sequências genômicas do Brasil .

index	name	33	MZ419872.1	66	MZ477800.1	99	MZ477842.1
0	MZ169910.1	34	MZ419873.1	67	MZ477801.1	100	MZ477843.1
1	MZ169911.1	35	MZ419874.1	68	MZ477804.1	101	MZ477846.1
2	MZ169912.1	36	MZ419875.1	69	MZ477805.1	102	MZ477847.1
3	MZ169913.1	37	MT807936.1	70	MZ477806.1	103	MZ477848.1
4	MZ169914.1	38	MW593153.1	71	MZ477807.1	104	MZ477849.1
5	MZ169915.1	39	MW592707.1	72	MZ477810.1	105	MZ477851.1
6	MZ169916.1	40	MW962308.1	73	MZ477811.1	106	MZ477852.1
7	MZ397170.1	41	MW962307.1	74	MZ477813.1	107	MZ477853.1
8	MZ169917.1	42	MZ477745.1	75	MZ477814.1	108	MZ477854.1
9	MZ397163.1	43	MZ477746.1	76	MZ477815.1	109	MZ477855.1
10	MZ397164.1	44	MZ477748.1	77	MZ477816.1	110	MZ477856.1
11	MZ397166.1	45	MZ477751.1	78	MZ477817.1	111	MZ477857.1
12	MZ397167.1	46	MZ477753.1	79	MZ477818.1	112	MZ477858.1
13	MZ397161.1	47	MZ477754.1	80	MZ477819.1	113	MZ477859.1
14	MZ264787.1	48	MZ477755.1	81	MZ477820.1	114	MT754277.1
15	MT738101.1	49	MZ477756.1	82	MZ477821.1	115	MT827075.1
16	MT738173.1	50	MZ477757.1	83	MZ477822.1	116	MT827190.1
17	MZ419856.1	51	MZ477758.1	84	MZ477824.1	117	MT827872.1
18	MZ419857.1	52	MZ477762.1	85	MZ477825.1	118	MT827940.1
19	MZ419858.1	53	MZ477764.1	86	MZ477827.1	119	MT827202.1
20	MZ419859.1	54	MZ477769.1	87	MZ477828.1	120	MT835026.1
21	MZ419860.1	55	MZ477786.1	88	MZ477829.1	121	MT835027.1
22	MZ419861.1	56	MZ477787.1	89	MZ477830.1	122	MT844030.1
23	MZ419862.1	57	MZ477788.1	90	MZ477831.1	123	MT835383.1
24	MZ419863.1	58	MZ477790.1	91	MZ477832.1	124	MT827074.1
25	MZ419864.1	59	MZ477792.1	92	MZ477833.1	125	MT710714.1
26	MZ419865.1	60	MZ477793.1	93	MZ477834.1	126	MT939524.1
27	MZ419866.1	61	MZ477795.1	94	MZ477835.1	127	MT939859.1
28	MZ419867.1	62	MZ477796.1	95	MZ477837.1	128	MT939654.1
29	MZ419868.1	63	MZ477797.1	96	MZ477838.1	129	MT126808.1
30	MZ419869.1	64	MZ477798.1	97	MZ477840.1	130	MT350282.1
31	MZ419870.1	65	MZ477799.1	98	MZ477841.1	131	MZ191508.1
32	MZ419871.1						

Fonte: (NCBI, 2022a)